

METHOD AND APPARATUS FOR TRANSFERRING DATA FROM FIRST PROTOCOL TO A SECOND PROTOCOL

Publication number: JP2002512766 (T)

Publication date: 2002-04-23

Inventor(s):

Applicant(s):

Classification:

- international: G06F13/12; G06F13/36; H04L12/56; H04L29/06; G06F13/12; G06F13/36; H04L12/56; H04L29/06; (IPC-1-7): G06F13/12; G06F13/36; H04L29/06

- European: H04L12/56(1); H04L29/06

Application number: JP19990550553T 19990329

Priority number(s): US19980054649 19980403; WO1999US06772 19990329

Also published as:

WO9952253 (A1)
US6185620 (B1)
EP0998886 (A1)

Abstract not available for JP 2002512766 (T)

Abstract of corresponding document: **WO 9952253 (A1)**

A method and apparatus for transferring data from a host to a node through a fabric connecting the host to the node. A chip architecture is provided in which a protocol engine provides for on chip processing in transferring data such that frequent interrupts from various components within the chip may be processed without intervention from the host processor. Additionally, context managers are provided to transmit and receive data. The protocol engine creates a list of transmit activities, which is traversed by the context managers, which in turn execute the listed activity in a fashion independent from the protocol engine. In receiving data, the context managers provide a mechanism to process frames of data originating from various sources without requiring intervention from the protocol engine. When receiving data, the context managers are able to process frames from different sources, which arrive out of order. Additionally, the context managers also determine when all frames within a sequence have been received. A link control unit is provided in which loop management is provided when the host is connected to a loop. Management of the loop includes implementing mechanisms to initiate acquisition of the loop and initiate a release of the loop in response to conditions in which data is received and transmitted by the host and by other nodes on the loop.



Data supplied from the **espacenet** database — Worldwide

| (51) Int.Cl. ⁷ | 識別記号 | F I | テーマコード (参考) |
|---------------------------|-------|---------------|-------------|
| H 0 4 L 29/06 | | G 0 6 F 13/12 | 3 3 0 A |
| G 0 6 F 13/12 | 3 3 0 | 13/36 | 3 2 0 A |
| 13/36 | 3 2 0 | H 0 4 L 13/00 | 3 0 5 B |

審査請求 未請求 予備審査請求 有 (全 73 頁)

(21) 出願番号 特願平11-550553
 (86) (22) 出願日 平成11年3月29日 (1999.3.29)
 (85) 優先文提出日 平成11年12月3日 (1999.12.3)
 (86) 国際出願番号 P C T / U S 9 9 / 0 6 7 7 2
 (87) 国際公開番号 W O 9 9 / 5 2 2 5 3
 (87) 国際公開日 平成11年10月14日 (1999.10.14)
 (31) 優先権主張番号 0 9 / 0 5 4 , 8 4 9
 (32) 優先日 平成10年4月3日 (1998.4.3)
 (33) 優先権主張国 米国 (U S)
 (81) 指定国 E P (A T, B E, C H, C Y, D E, D K, E S, F I, F R, G B, G R, I E, I T, L U, M C, N L, P T, S E), J P

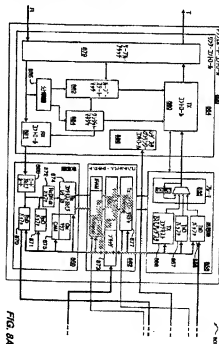
(71) 出願人 エルエスアイ ロジック コーポレーション
 アメリカ合衆国カリフォルニア州95035,
 ミルピタス, マッカーシー・ブルバード
 1551
 (72) 発明者 ウェーバー, デーヴィッド・エム
 アメリカ合衆国コロラド州80132, モニュ
 メント, レインジ・ビュロー・ロード 3680
 (72) 発明者 ホーグランド, ティモシー・イー
 アメリカ合衆国コロラド州80918, コロラ
 ド・スプリングズ, パーグトリイ・ドライ
 ブ 3035
 (74) 代理人 弁理士 社本 一夫 (外5名)

最終頁に続く

(54) 【発明の名称】 第1のプロトコルから第2のプロトコルへのデータ転送方法及び装置

(57) 【要約】

ホストをノードに接続するファブリックを介してデータをホストからノードまで転送する方法及び装置である。チップ・アーキテクチャが提供されており、そこでは、データの転送の際にプロトコル・エンジンがオンチップ処理を提供し、チップ内の様々な要素からの煩雑な割り込みが、ホスト・プロセッサからの介入なしに処理される。更に、データの送受信のためのコンテキスト・マネージャが提供される。プロトコル・エンジンは、送信アクティビティのリストを作成し、このリストはコンテキスト・マネージャによって検討され、コンテキスト・マネージャは、リスト化されたアクティビティをプロトコル・エンジンとは独立した態様で実行する。データを受信する際には、コンテキスト・マネージャは、様々なソースからのデータのフレームを、プロトコル・エンジンからの介入を要求することなく処理するメカニズムを提供する。データを受信するときには、コンテキスト・マネージャは、異なるソースから順不同で到着するフレームを処理することができる。更に、コンテキスト・マネージャは、あるシーケンス内のすべてのフレームがいつ受信されたかを判



【特許請求の範囲】

1. チップであって、
入力ポートと、
出力ポートと、
第1の処理ユニットであって、

このチップから離れて位置するホストからのリクエストを受信し、データを宛先に送信する第1の受信手段と、

前記データを受信する第2の受信手段と、

前記宛先と前記データとを、前記宛先への送信に先だって、このチップに接続されたメモリに記憶する記憶手段と、

を備えている第1の処理ユニットと、

宛先に送信するデータの存在を検出する検出手段を含む第2の処理ユニットと

、
前記データと前記宛先とを用いて前記宛先への搬送のためのフォーマットに前記データをフォーマットするフォーマット手段と、

を備えていることを特徴とするチップ。

2. 請求項1記載のチップにおいて、前記第1の処理ユニットは、宛先に送られるデータののためのアクティビティのリストを作成し、前記検出手段は、前記リストを検査し前記宛先に送信するためのデータの存在を検出することを特徴とするチップ。

3. 請求項1記載のチップにおいて、予測されるデータのリストが前記第1の処理ユニットによって発生され、前記第2の処理ユニットは、

データを遠隔ソースから受信する受信手段と、

前記データを記憶する識別手段と、

前記第1の処理ユニットに、いつすべてのデータが前記遠隔ソースから受信されたかを指示する指示手段と、

を含むことを特徴とするチップ。

4. 請求項1記載のチップにおいて、前記第2の処理ユニットは埋め込み型プロセッサであることを特徴とするチップ。

5. 請求項1記載のチップにおいて、前記第2の処理ユニットはステート・マシンであることを特徴とするチップ。

【発明の詳細な説明】**第1のプロトコルから第2のプロトコルへのデータ転送方法及び装置****関連出願への相互参照**

この出願は、"Method and Apparatus for Managing Access to a Loop in a Data Processing System"と題し、出願番号が09/054,850であって、弁護士的事件（ドケット）番号が98-090である、同じ被譲渡人に譲渡されている同時出願中の出願と関連する。この関連出願は、本出願において援用することとする。

1. 技術分野

本発明は、広くは、改善されたデータ処理システムに関し、詳しくは、1つのデータ・プロトコルから別のデータ・プロトコルにデータを転送する改善された方法及び装置に関する。更に詳しくは、本発明は、ファイバ・チャネルなどのような、チャネル・プロトコルからシリアル・プロトコルにデータを転送する改善された方法及び装置を提供する。

2. 関連技術の説明

ANSIによって採用されたファイバ・チャネル・スタンダード（FCS）は、ワークステーション、大容量記憶装置、プリンタ、ディスプレイなどに対する低コストで高速の相互接続のためのスタンダードを提供する。ファイバ・チャネル（FC）は、分散型のシステム・アーキテクチャ、イメージ集約的なローカル・ネットワーク及びクラスタにとって理想的である。ファイバ・チャネルは、媒体依存的であり、マルチ・ベンダ相互運用性（インターオペラビリティ）を提供する。

現在のファイバ・チャネルでのデータ転送速度は、それぞれの方向で100メガバイト／秒を超えている。また、ファイバ・チャネルのデータ転送速度は、50メガバイト／秒や25メガバイト／秒程度のより低速にスケーリングすること

ができる。この技術によると、切り換えられる媒体と共有される媒体との両方に対するチャネル接続とネットワーク接続との両方をサポートするインターフェースが得られる。ファイバ・チャネルは、デバイスの相互接続を単純化し、ハードウェアのコストを減少させる。その理由は、それぞれのデバイスが、チャネル・

インターフェースとネットワーク・インターフェースとの両方に対して、ただ1つのファイバ・チャネルだけしか要求しないからである。ネットワーク、ポート間及び周辺機器のインターフェースは、任意のフォーマットのデータ転送との同じハードウェア接続を介してのアクセスが可能である。

ターゲットとソースとの間でのデータ転送に関し、入出力（I/O）プロセスサ技術の性能が急激に向上したために、より高速で、より多くの接続性を提供し、より長い距離にわたる接続を可能にするI/Oソリューションを求めて、高性能サーバ、ワークステーション、クラスタ型計算及び関連するストレージ市場に対する要求が非常に大きくなった。大容量ストレージと全二重ファイバ・チャネル・リンク上のそれ以外のプロトコルとをサポートするように設計されている高性能でインテリジェントなI/Oプロセッサであるファイバ・チャネルI/Oプロセッサは、I/O動作をサポートするのに要求されるホストCPU及びPCI帯域幅を減少させる態様でデータを移動させることが求められる。初期化、コマンド、エラー回復など、データの移動ではないアクティビティのためにPCIバスなどのシステム・バス上で費やされる時間の長さを最小化することが求められる。従って、2つの異なるデータ・プロトコルの間でデータを転送する方法及び装置の改良が求められている。

3. 発明の概要

本発明は、ホストから第1のノードまでホストを第1のノードに接続するバスを介し、更に、ファブリックを介して第1のノードに接続された第2のノードまでデータを転送する方法及び装置を提供する。第1のノードは、チップ・アーキテクチャを含んでおり、そこでは、データの転送の際にプロトコル・エンジンがオンチップ処理を提供し、チップ内の様々な要素からの頻繁な割り込みがホスト・プロセッサからの介入なしに処理されるようになっている。更に、データ

の送受信のためのコンテキスト・マネージャが提供される。プロトコル・エンジンは、送信アクティビティのリストを作成し、このリストはコンテキスト・マネージャによって検討され、コンテキスト・マネージャは、リスト化されたアクティビティをプロトコル・エンジンとは独立に実行する。データを受信する際には、コンテキ

スト・マネージャは、様々なソースからのデータのフレームを、プロトコル・エンジンからの介入を要求することなく処理するメカニズムを提供する。データを受信するときには、コンテキスト・マネージャは、異なるソースから順不同で到着するフレームを処理することができる。更に、コンテキスト・マネージャは、あるシーケンス内のすべてのフレームがいつ受信されたかを判断する。

更に、本発明は、ホストがループに接続される時にループ管理が提供されるリンク制御（コントロール）ユニットを提供する。ループの管理には、ホストとループ上の他のノードとによってデータが受信及び送信される状態にตอบสนองして、ループの取得を開始しループの解放を開始するメカニズムの実現が含まれる。

4. 図面の簡単な説明

本発明の特徴付けると考えられる新規な特徴は、請求の範囲に記載されている。しかし、本発明自体は、その使用、更なる目的及び効果の好適な態様と共に、実施例に関する以下の詳細な説明を次の添付の図面と共に参照することによって、最もよく理解されるはずである。

図1は、本発明の好適実施例によるファイバ・チャネルにおける5つの層を図解している。

図2A及び2Bは、リンク制御フレームとデータ・フレームとである。

図3は、エクスチェンジ（exchange）の図と、それがいかにしてその最小の要素に分解されるかを示している。

図4は、本発明の好適実施例によって処理されうるSCSIエクスチェンジを図解している。

図5は、本発明が実現されうるデータ処理システムを示している。

図6は、本発明の好適実施例によるデータ処理システムのブロック図である。

図7は、本発明の好適実施例によるメッセージ要求処理を図解する図である。

図8は、本発明の好適実施例が実現されうるチップのブロック図である。

図9は、本発明のシステム内でのデータ転送を図解する機能ブロック図である。

図10は、本発明の好適実施例によるFree_List循環キュー（待ち行列）とPos

l_List循環キューとを図解している。

図11は、本発明の好適実施例による送信コンテキスト・マネージャにおいて実現されるプロセスを図解する流れ図である。

図12は、本発明の好適実施例による受信制御ブロックのためのフォーマットである。

図13は、本発明の好適実施例によってコンテキスト・スイッチングを実行するのに用いられる流れ図である。

図14は、本発明の好適実施例によるDMA開始プロセスの流れ図である。

図15は、本発明の好適実施例によるDMA更新プロセスの流れ図である。

図16は、本発明の好適実施例によるフレーム完成処理プロセスである。

図17は、本発明の好適実施例によるループ管理制御のためのステート・マシンである。

図18は、本発明の好適実施例によって解放状態(open state)のループを管理するプロセスの流れ図である。

図19は、本発明の好適実施例によってアイドル状態でループの取得を制御するのに用いられる規則を組み入れているプロセスの流れ図である。

図20は、本発明の好適実施例によるループ状態待機に用いられる規則を組み入れているプロセスの流れ図である。

図21は、本発明の好適実施例による判断ウィンドウ状態における変化を扱う際に用いられる規則を組み入れているプロセスの流れ図である。

図22は、本発明の好適実施例による判断ウィンドウ状態における変化を扱うための規則を組み入れているプロセスの流れ図である。

5. 詳細な説明

ファイバ・チャネルは、インターネット・プロトコル(IP)などのネットワーク・プロトコルやSCSIなどのチャネル・プロトコルをサポートする高

性能リンクである。ファイバ・チャネルの構造は、5つの層によって定義される。図1は、ファイバ・チャネルにおける5つの層を図解している。最下位の層であるFC-0は、媒体インターフェース層である。この層は、2つのデバイス間

の物理的インターフェースを定義する。この層には、ドライバと、受信機と、銅から光へのトランスデューサと、コネクタと、銅又は光ケーブル上を133、266、531、1062メガバイト／秒の速度で送信又は受信するのに必要な任意の他の低レベル関連回路とが含まれる。

1つ上の次の層は、FC-1層である。この層は、8b/10bの符号化／復号化と、データ統合に必要な送信プロトコルと、送信クロック及び受信クロック回復とを定義している。この層は、通常、FC-0層とFC-2層とを実現しているハードウェアの間に分割される。特に、FC-0トランシーバは、クロック回復回路を含み、他方で、8b/10bの符号化／復号化は、FC-2の層でなされる。次の層は、FC-2層である。この層は、フレーミング・プロトコルと転送されているデータに対するフロー制御動作とを定義する。送信又は受信されているデータの意味は、FC-2に対して透過的 (transparent) である。しかし、任意の与えられたフレームの組のコンテキストは、FC-2において維持される。フレーミング・プロトコルは、必要なフレームを作成し、データは、それぞれのフレームのペイロード内にパケット化される。次の層は、FC-3層である。FC-3は、複数のN_portにわたる共通のサービスを提供する。N_portは、「ノード」ポートとも称されるが、リンクのノード端部におけるファイバ・チャネルで定義されるハードウェア・エンティティ (entity) である。これらのサービスには、ストリッピング (Striping)、ハント・グループ (Hunt Groups) 及びマルチキャスト (Multicasting) が含まれる。これらのサービスすべてにより、1つのポートが複数のN_portと一度に通信することが可能になる。FCにおいて定義される最上位の層は、FC-4層である。FC-4層は、既存のスタンダードのシームレスな統合を提供する。この層は、上位の層のプロトコル (ULP) から下位の層へのマップ (mapping) を特定する。これらのULPのいくつかには、SCSIとインターネット・プロトコル (IP) とが含まれる。これらのULPのそれぞれは、それ自身のANSIドキュメントにおいて定義さ

れる。

ファイバ・チャネルにおいて用いられるフレームには2つのタイプがある。す

なわち、リンク制御フレームと、データ・フレームとである。リンク制御フレームは、ペイロード (payload) を含まず、データ・フレームへの応答である。データ・フレームは、ペイロード・フィールド内のデータを含むフレームである。図2A及び2Bを参照すると、リンク制御フレーム200とデータ・フレーム202とが図解されている。それぞれのフレームは、フレームの開始 (start-of-frame=SOF) フィールド204を含み、フレームの終了(end-of-frame=EOF) フィールド206の順序付き集合 (ordered set) で終わる。SOFとEOFを含むすべての順序付き集合は、4バイトで構成される。それぞれのフレームは、宛先 (destination) 及びソースID、サービスのクラス、フレームのタイプ (すなわち、SCSI又はIP) などの事項を定義する24バイトのヘッダ・フィールド208を少なくとも含む。フレーム内の最大のフィールドは、データ・フレーム202内にあるペイロード・フィールド210である。フレームは、リンク制御フレームであるならばペイロード・フィールド210は存在せず、データ・フレームであるならば212バイトまでのペイロード・フィールドを含む。最後に、両方のタイプのフレーム共に、送信エラーの検出に用いられる周期上長検査 (CRC) フィールド212を含む。

ファイバ・チャネルにおいて用いられるそれ以外の構成は、シーケンスとエクステンションとである。次に図3を参照すると、エクステンションの図とそれがどのようにして最小の要素に分解されるかが示されている。エクステンション300には、シーケンス302などの1つ又は複数のシーケンスが含まれる。それぞれのシーケンスは、フレーム304などの1つ又は複数のフレームから構成される。エクステンションは、典型的なSCSI/Oを考えることによって最もよく図解される。SCSI/Oでは、複数のフェーズが存在し、これらがI/Oを構成する。これらのフェーズには、コマンド、データ、メッセージ及びステータスのフェーズが含まれる。

SCSI (FCP) ULPのためのファイバ・チャネル・プロトコルを用いると、これらのフェーズは、残りの下位のFC層にマップされうる。図4は、本発

明の好適実施例によって扱われうるSCSIエクステンションを図解している。S

C S I エクスチェンジ400は、コマンド・シーケンスであるC M D S E Q 402と、データ・リクエスト・シーケンスであるD A T A R E Q S E Q 404と、データ・シーケンスであるD A T A S E Q 406と、応答シーケンスであるR S P S E Q 408とを含む。

図5には、本発明が実現され得るデータ処理システムが示されている。データ処理システム500は、ファブリック506を介してターゲットに接続されたイニシエータ502を含む。示されている例におけるファブリック506は、ポイント間(point-to-point)、切換型(switched)及び仲裁型(arbitrated)ループを含む様々な幾何学的形態(トポロジ)をもちうるファイバ・チャネル・ファブリックである。データ処理システム500では、エクスチェンジのフローは、ターゲット504へのフレームを1つ含むコマンド・シーケンスであるC M D S E Q 402をイニシエータ502が送出することと共に開始する。このフレーム内のペイロードは、コマンド記述子ブロック(C D B)を含む。ターゲット504は、フレームを1つ含むデータ配送リクエスト・シーケンスD A T A R E Q S E Q 404に応答する。このフレームのペイロードは、転送準備完了応答(transf er ready response)を含む。イニシエータ502は、いったんこの応答を受け取ると、1つ又は複数のフレームを含むデータ・シーケンスD A T A S E Q 406の送出を開始する(D A T A O U T フェーズ)。ターゲットは、最後のフレームを受け取ると、フレームを1つ含む応答シーケンスR S P S E Q 408を送出する。このシーケンスによって、S C S I エクスチェンジは終了する。

本発明は、異なるデータ・プロトコルの間でデータを転送するシステム、アーキテクチャ及び方法を提供する。示されている例は、S C S I プロトコルとファイバ・チャネル・プロトコルとの間でのデータ転送のためのものである。本発明は、ホストを高速ファイバ・チャネル・インターフェースに接続するのに用いられ、また、切換型ファブリック、ポイント間、そして最も重要な仲裁型のループを含むすべてのファイバ・チャネル・トポロジにおいて用いられる。

次に、図面を、特に、図6を参照すると、本発明の好適実施例によるデータ処理システムのブロック図が示されている。データ処理システム600は、ホスト

602を含み、このホストは、データ処理システム600の1つ又は複数のCPUを形成する1つ又は複数のプロセッサを含みうる。データ処理システム600は、I/Oスペシャル・インタレスト・グループから入手可能なインテリジェント入出力(I/O)アーキテクチャの仕様(1997年3月)と共に設計されたデータ処理システムである。これは、本出願において援用する。しかし、本発明は、他のシステム・アーキテクチャを用いても実現することができる。

ホスト602の中のプロセッサは、例えば、300MHzで動作するペンティアムIIプロセッサである。このプロセッサは、米国カリフォルニア州サンタクララ所在のインテル社から入手可能である。示されている例では、一次バス604と二次バス606とはPCIバスであるが、本発明は、他のタイプのバスを用いても実現が可能である。

更に図6を参照すると、データ処理システム600は、一次入出力プラットフォーム(IOP)608を含む。これは、一次バス604を介して、ホスト602に接続されている。更に、IOP608は、二次バス606に接続され、PCI・PCI間のバス・ブリッジとして機能する。データ処理システム600はまた、アダプタ612及び614を含む。二次IOP610及び616は、I/Oの下にあるインテリジェント・アダプタであり、二次IOP610及び616は、入出力プロセッサを含む。アダプタ612及び614は、非インテリジェントなアダプタであり、入出力プロセッサを含まない。

本発明のシステムは、ホストからチップヘリクエスト・メッセージを転送する、そして、逆にチップからホストにリプライ・メッセージを転送するメカニズムとして、リクエスト及びリプライ・メッセージ・キューを用いる。リクエストは、ホストからチップを介してデバイスまでの経路を表し、他方で、リプライは、デバイスからチップを介してホストまでの経路を表す。

リクエスト及びリプライ・メッセージ・キューは、メッセージ・フレームの予め配分されたリストであり、共有される又はホストのメモリに存在する。チップの内部的には、それぞれのキューは、それぞれが予め配分されたメッセージ・プール内にメッセージ・フレームのアドレスを含む2つのFIFOであるフリー・リストとポスト・リストとによって特徴付けられる。フリー及びポスト・リスト

は、ホストからは可視的ではないが、メッセージ・プール内で自由な及びポストされたメッセージを管理する際にチップをサポートする。

チップが初期化されるときには、ホストは、リクエスト及びリプライ・キューがどのように管理されるべきかを選択する。デフォルトでは、リクエスト・キューは、ホスト・メモリに存在しうる。オプションとして、リクエスト・キューは、ホストとチップとの間で共有されるメモリに存在する。リプライ・キューは、常に、ホスト・メモリに存在する。リクエスト・キューとリプライ・キューとの両方へのアクセスは、P C I アドレス空間にマップされた2つのレジスタを介して提供される。

図7を参照すると、本発明の好適実施例によるメッセージ・リクエスト処理を図解している図が示されている。ホスト700は、メッセージを構築して、リクエスト／リプライ・レジスタ702を読み出し次の空のメッセージ・フレームのアドレスをフリーFIFO704におけるメッセージ・フレーム・プールから検索することによって、フリー・メッセージ・フレームを配分する。次に、ホスト700は、そのリクエストをリクエスト・キュー706におけるメッセージ・フレームに書き込む。その後で、ホスト700は、フレームのアドレスをリクエスト／リプライ・レジスタ702に書き込み、このレジスタは、このリクエストをサービスのためにポストFIFO710におけるチップ708にポストする。ホスト700は、次に、利用可能なフリー・メッセージが尽きるまで、このプロセスを反復して、より多くのリクエストをポストする。チップ708は、リクエストのアドレスをリクエスト／リプライ・レジスタ702から読み出し、リクエスト・キュー706の中のアドレスにおけるメッセージを処理し、メッセージ・アドレス（この時点では、空のメッセージ・フレーム）を再びリクエスト／リプライ・レジスタ702に書き込むことによって、ポストされたリクエストを読み出す。ホスト700がリクエスト／リプライ・レジスタ702を読み出すときにフリー・メッセージ・フレームが存在しない場合には、チップ708によって供給される値は、示されている例では、FFFF-FFFFhである。

リプライ・キュー712は、リクエスト・キュー706の場合と似た態様で管理される。ただし、この場合には、チップ708が作成側（producer）となる

点が異なる。ホスト700は、リプライ・キュー712においてリプライ・メッセージ・プールを配分し、それぞれのメッセージ・フレームのアドレスをリプライ・レジスタ714にポストする責任を有する。チップ708は、リプライを送出することを望むときには、フリーFIFO704における次のフリー・メッセージ・フレームのアドレスを読み出す。チップ708は、リプライ・メッセージ・キュー712におけるフレームをメッセージで満たし、フレームのアドレスをリクエスト／リプライ・レジスタ702にポストし、リクエスト／リプライ・レジスタ702は、このアドレスをポストFIFO710に書き込む。チップ708は、このプロセスを反復することによって、複数のリプライをポストすることがある。ホスト700は、リクエスト／リプライ・レジスタ702を読み出し、ポストされたリプライ・メッセージのアドレスをポストFIFO710から検索する。ホストは、このメッセージをいったん消費すると、アドレス（この時点では、フリー・メッセージ・フレーム）をリクエスト／リプライ・レジスタ702に書き込み、リクエスト／リプライ・レジスタ702は、このアドレスをフリーFIFO704に書き込む。ホスト700がリクエスト／リプライ・レジスタ702を読み出すときにポストされたメッセージが存在しないときには、ホスト700は、示されている例では、FFFF-FFFFhの値を受け取る。

本発明は、リクエスト及びリプライ・キューを用いて、リクエスト及びリプライをホスト・ドライバとチップとの間で転送する。ホストがこれらのキューと相互作用する態様は、パフォーマンスに影響しうる。本発明の好適実施例によって用いられるメッセージ・キューイングには、2つのモデルが存在する。データ転送のための「プッシュ・プッシュ」モデルは、リクエスト・キューをチップによって提供されるものと定義し、リプライ・キュー・メモリをホスト・メモリ内に常駐するものとして定義する。このモデルは、データをキューの中に「プッシュ」する（押し込む）ために、リクエスト又はリプライのどちらかのイニシエータを必要とする。多くの状況で、このモデルは最適ではない。

データ転送の「プル・プッシュ」モデルでは、リクエスト及びリプライ・キューがホスト・メモリ内に存在することが要求される。リクエストは、バス・マスタモードで動作しているチップの中に「プル」され（引き込まれ）、他方で、リプライ

は、ホスト・メモリの中に「プッシュ」される(押し込まれる)。このモデルによれば、チップがすべてのキューイングのためにホスト・メモリを用いることが可能となる。また、このモデルによると、チップがその動作を合理化することも可能となる。その理由は、このモデルによれば、プッシュ・プッシュ・モデルの場合のように突然に作用することを強制されるのではなく、いつコマンドを処理することを望むのかを決定することができるからである。このオプションは、複数のブリッジを介して直接にP C Iバスにアクセスすることにより過剰なプロセッサ・オーバーヘッドを生じるようなホスト環境に最も適している。

両方の動作モード共に、P C Iバスへの同じ数のアクセスを必要とし、同じレジスタ・セットを介するキュー・アクセスを提供する。チップのためのデフォルトのオプションは、「プッシュ・プッシュ」モデルである。「プル・プッシュ」モデルは、メッセージによって呼び出すことができる。

次に、図8を参照すると、本発明の好適実施例を実現することができるチップのブロック図が図解されている。チップ800は、プロトコル・エンジン802と、データ移動ユニット804と、移動制御ユニット806とを含む。

プロトコル・エンジン802は、この分野の当業者に既知である多数の異なるタイプのプロトコル・エンジンを用いて実現され得る。示されている例では、プロトコル・エンジン・コア807は、32ビットのRISCコア808に基づいている。RISCコア808は、20-30MIPSの性能を有している。プロトコル・エンジン802は、それ自身のコントローラ812とリード/フェッチ/ライト・ユニット814とを備えた埋め込み型モジュール・バス(EMB)810を含む。EMB810は、バス上でのモジュール相互間の通信のための標準化されたモジュール・インターフェースを提供する。EMB810は、また、示されている例では、複数のバス・マスタをサポートしている。

示されている例では、プロトコル・エンジン・コア807は、8KBの命令/データ・バッファ816を含み、限界コード(critical code)及びデータ構造に対するゼロ待機状態のスタティックRAM領域を提供する。割り込みコントローラ818とクロック/リセット・コントローラ820とが、プロトコル・エンジン・コア807の中に見出される。RISCコア808は、EMB810に

接続されたコントローラ822を用いている。

プロトコル・エンジン802の中のシステム・インターフェース824は、コンフィギュレーション及び高順位コマンドと、パケット化されたリクエストと、ホストとチップ800との間のリプライ・メッセージとをサポートする。システム・インターフェース824は、非データ転送のためのPCIバス・トラフィックを最小化するように設計されている。システム・インターフェース824は、また、I/Oリクエストとリプライ・メッセージ・パケットとをホスト・メモリとチップ800との間で転送するのにも用いられる。DMA FIFO828を備えたマスタ制御826とスレーブFIFO832を備えたスレーブ制御830とは、システム・インターフェース824の内部に存在する。EMBアタッチ834は、システム・インターフェース824のために、EMBバス810への接続を提供する。DMA及びSRW制御836と、スレーブ・アクセス制御838と、メッセージFIFO840とは、システム・インターフェース824の内部に存在する。マスタ制御及びスレーブ制御ユニットは、ホスト・インターフェースとDMA FIFO及びスレーブFIFOそれぞれとの間のデータ転送を提供する。これらのユニットは、システム・メモリへの/システム・メモリからのデータを（バス・インターフェース・ユニットを介して）これらのFIFOの中へ及びFIFOから外へバーストする。これらのFIFOからのデータは、次に、EMBアタッチ機能（DMA FIFO）又はスレーブ・アクセス制御機能（スレーブFIFO）を介して、ローカル・メモリに、及び、ローカル・メモリから移動される。

DMA及びSRWユニットは、バースト転送（DMA）又はシングル・サイクル転送（SRW）に対するデータ転送を規制する。msg FIFO制御ユニットは、個々のキュー要素をローカル・メモリに、及び、ローカル・メモリからリード/ライトすることを含むメッセージング・キューを実現するのに必要なハードウェアを提供する。示されているシステム・インターフェースは、この分野の当業者に既知である多数の方法で実現することができ、プロトコル・エンジン802と示されているものとは別のデータ移動ユニット804との間のインターフェースが提供される。

プロトコル・エンジン802は、チップ800の外部にあるメモリへの接続を提供する外部メモリ・コントローラ842を含む。メモリ・コントローラ842は、32ビットのプラス・パリティDRAM（ファースト・ページ及びEDO）と、EPROMと、フラッシュ（8ビット）及びシリアルEEPROMとをサポートする。示されている例では、データを転送するメモリ・データ経路844は、DRAM制御846とフラッシュ制御848とによって制御される。最後に、プロトコル・エンジン802は、EMBレジスタ及びタイマ・ユニット850に位置する自由動作（free-running）タイマを含む。このタイマは、イベント・タイム・スタンピングに用いることができる。

示されている例では、英国ケンブリッジ所在であり米国テキサス州オースチンにもオフィスを有するアドバンスドRISCマシン社（Advanced RISC Machine Ltd.）から市販されているRISCコア808用のARMマイクロプロセッサ・コアを用いている。

データ移動ユニット804は、バス・インターフェース852と、送信機DMAユニット854と、受信機DMAユニット856とを含む。データ移動ユニット804の内部の要素は、既知のバス・インターフェース・ユニットと、送信機DMAユニットと、受信機DMAユニットとを用いて実現することができる。送信機DMAユニット854と受信機DMAユニット856とは、それぞれ、送信データ転送エンジンと受信データ転送エンジンとも称される。バス・インターフェース・ユニット852は、示されている例では、PCIバスを介して情報を通過させるのに用いられる。その際に、インターフェース・ユニットは、PCIマスタ・インターフェース859とPCIスレーブ・インターフェース862とを含む普遍的（universal）なPCIインターフェース858を介してマスタ及びスレーブ両方のPCIバス・サイクルをサポートする。バス・インターフェース・ユニット852の中には、アービタ（仲裁、arbiter）及びキャッシュ・サイクル・コントローラ863があり、プログラム可能なアービタが、プロトコル・エンジン802と送信機DMAユニット854と受信DMAユニット856との間での仲裁を行うのに用いられる。データ移動ユニット804は、送信機DMAユニット854における送信機バッファに向かう、又は、受信DMAユニット8

6における受信バッファから出る、複数の散乱・集合（scatter-gather）データ・エントリを整列させる（align）ように設計されている。この整列（アライメント）は、32ビットのダブル・ワード境界上でのものである。結果的に、入ってくるどれかのデータが奇数のカウントを有している場合には、フィル・バイトが追加される。データ移動ユニットは、送信機DMAユニット854における1つの送信散乱／集合（S/G）FIFOと、受信DMAユニット856における2つの受信散乱／集合（S/G）FIFOとを含む。3つのFIFOのすべてが、3つのS/Gエントリを含む。2つのエントリは、現在のS/Gエントリであり、残りのエントリは、次の処理されるべきS/Gエントリである。データ移動ユニット804は、既知のDMAチャネル設計を用いて実現され得る。

移動制御ユニット806は、送信機858と、受信機860と、コンテキスト・マネージャ862と、リンク制御ユニット864とを含む。送信機858の内部には、TxDPバッファ866とTxOPバッファ867と、送信コンテキスト及びレジスタ868とへの入力を備えたフレーマ865がある。受信機860は、リンク制御ユニット864に接続されたロード及びルート・ユニット869を含む。受信機860の中には、RxDPバッファ872と、CtxマシNCAM873と、受信コンテキスト及びレジスタ874とも存在する。コンテキスト・マネージャ862は、コンテキスト・マネージャ862をEMB10に接続するブリッジ875と、送信機858と、受信機860とを含む。マイクロコード・エンジン876は、送信コンテキスト・マネージャ877と受信コンテキスト・マネージャ878とを制御する。コンテキスト・マネージャ862は、プロトコル・エンジン802をフリー・アップ（free up）して他の機能を実行するデータ転送機能を提供する。送信機858は、状態及びコンフィギュレーションのためのレジスタ（送信機コンテキスト及びレジスタ868）と、データ・ストレージ・バッファ（TxDPバッファ866及びレジスタ868）と、フレーマ865とを含む領域に対する機能を提供する。フレーマ865は、データをバッファから取り出し、コンフィギュレーション・レジスタからの任意の要求される情報を加えて、法的

(legal) なファイバ・チャネル・フレームを発生する責任を有する。この情報は、SOFと、ヘッダと、ペイロードと、CRCと、EOFとを含む。フレーム

865は、次に、リンク制御ユニット864へのリクエストをアサートし、フレームを正しい宛先に送る。

フレーム865は、フレームの全体が送られるまでリンク制御ユニット864のリクエストに基づいてデータを搬送する。フレーム865は、また、データが2つのバッファであるTxDバッファ866及びTxOバッファ867から搬送する責任を有しており、それによって、バッファがロードされる。フレーム865のCRCサブブロックは、ファイバ・チャネルの仕様によって定義されるエラー・チェック・コードを計算する責任も有している。フレーム865は、計算されたコードをデータ・ストリームの正しい地点に挿入する。TxDバッファ866は、PCIバス及びデータ移動ユニット804を介してホストからロードされたデータを含む。TxOバッファ867は、プロトコル・エンジンを介してロードされたデータを含む。このバッファは、ヘッダとSOFとEOFとペイロードとを備え、予めフォーマットされたフレームを含む。

受信機860は、このノードにアドレス指定されたフレームを取り出し、そのフレームの正しさをチェックし、そのフレームを正しいメモリの宛先に配分することを助けるという責任を有する。ロード及びルート・ユニット869におけるロード及びルート機能は、ヘッダにおけるあるフィールドを解析して、データをどのバッファに向けて経路決定すべきかを判断する。この判断は、トラフィック・タイプに基づく(すなわち、SCSIコマンド、SCSIデータ、If)。送信側と同じように、RxOバッファ871は、プロトコル・エンジン802によって処理されるように決められたフレームを含む。やはり、フレーム全体(ヘッダ及びペイロード)が、RxOバッファ871に含まれる。

RxDバッファ870に向けられたフレームは、そのヘッダ情報が取り除かれ、別個のRxDHバッファ872に配置される。CTxマシンであるCAM873は、ヘッダ情報を用いてこのフレームがどのデータ転送に属するのかを判断する。このブロックでは、CRCもチェックされて、無効なCRCを伴うフレーム

は廃棄される。C T xマシンであるCAM 873は、最後のフレームからヘッダ情報を比較して、それがシーケンスの中の次のフレームであるかどうかを判断する。r x a q m eが適切なs / g エントリを備えたDMAチャネルを提供するように

と要求される場合には、DMAチャネルは、データをバッファから除去する。R x D H バッファ 872 が別個のバッファであり、それによって、先のフレームが除去されているときに後のフレームからのヘッダを解析することができる。

リンク制御ユニット864は、ケーブル・アタッチ879を含むが、これは、ファイバ・チャンネルへの接続を提供し、ファイバ・チャンネルからのデータを送信及び受信する。送信(TX)制御ユニット880と受信制御ユニット881とは、リンク制御ユニット864において見出される。TX制御ユニット880は、送信機858からのデータを受信し、そのデータをケーブル・アタッチ879を介してファイバ・チャンネルの上に送る。TX制御ユニット880は、規則を適用して、フレームを送信することが望ましくかつ許容されるかを判断する。選択された時間においてフレームを転送することが望ましくなく、許容されない場合には、TX制御ユニット880は、フレームの送信を許容するのに必要な作用を実行する。ケーブル・アタック879は、8b / 10b符号化 / 復号化機能を提供し、選択された外部のシリアライザ / デシリアライザと互換な適切な態様でパイットを再度順序付ける。データは、受信制御ユニット881によって受信され、受信機860に向けて送信される。リンク制御ユニット864はまた、ループ・ステート・マシン882と、分類器(classifier)883と、クレジット・マネジャ884と、リンク制御レジスタ885とを含む。ループ・ステート・マシン882は、仲裁、送信及び受信プロトコルを含むループ関係機能を管理する。クレジット・マネジャ884は、フレーム・ベースのクレジット・プロトコルをモニタし管理する責任を有している。クレジット・マネジャ884は、別のノードがフレームをクレジット・マネジャ884が位置しているこのノードに送られることを可能にするにはいつクレジットが与えられるべきであるかについてトラッキングする。クレジット・マネジャ884は、また、フレームの送信を可能にする

のに十分なクレジットがいつ入手可能となるかについてもトラッキングする。任意の時点で、ノードは、その現在の宛先に送ることができる最大数のフレームを有する。これをクレジットと称する。ノードは、フレームを送信するときは常に、1クレジットを用いる。受信側のノードは、これらのフレームを限定されたバッファ・プールの中に受信する。フレームがバッファ・プールから除去されるとき

には、R_RDYと称されるプリミティブが発生される。送信側のノードは、R_RDYを受信するときには、そのクレジット・カウントをインクリメントする。

リンク制御レジスタ886は、リンク制御864のためのコンフィギュレーション及びステータス報告レジスタを含む。分類器885は、ループからの入来ワードをモニタし、多くのタイプのプリミティブの符号化を他のブロックに提供する。分類器885は、信号を、ループ状態マシン882とクレジット・マネージャ884とに提供する。分類器885におけるこの機能が提供されるのは、多くのブロックが同じプリミティブに反応からであり、復号の複写は、この状況では不要である。受信制御ユニット881はループ状態マシン882をモニタし、そのループ上を送信されているフレームはいつ受信制御ユニット881が位置しているノードに向けられるのかを決定する。以下でより詳細に説明するTX制御ユニット880を除き、リンク制御ユニット864内の要素は、米国規格協会(ANSI)からのF-CAL仕様を用いて、この分野の当業者に既知である要素を用いて実現することができる。

次に図9を参照すると、本発明のシステム内でのデータ転送を図解する機能ブロック図が示されている。図9は、本発明のシステム・アーキテクチャを構成する基本的な機能ブロックを示している。システム900の中には、2つの主要なグループにグループ分けできる複数の機能ブロックが示されている。すなわち、出ていく送信グループと入ってくる受信グループとである。ブロックの間の数字の付いた矢印は、本発明のアーキテクチャによって提供される機能におけるシーケンシャルなステップを表している。この図のそれぞれの側には、メモリ要素がある。すなわち、システム・メモリ902と、ローカル・メモリ904と、シス

テム・メモリ906と、ローカル・メモリ908とであり、これらは、リクエスト・メッセージ・フレームとリプライ・メッセージ・フレームとを管理するのに用いられるフリー循環キューとポスト・フリー循環キューとを含む。リクエスト・メッセージ・フレーム構造は、図9の左側のメモリであるローカル・メモリ904の中にあり、リプライ・メッセージ・フレーム構造は、右側のメモリであるローカル・メモリ908の中にある。図の中央には、送信及び受信データ経路があり、これらは、後で、図9を参照して更に詳細に説明される。オペレーティング・

システム・モジュール（OSM）910と、I/Oプラットフォーム（IOP）ドライバ912と、メッセージ・トランスポート・マネジャ914と、インターフェース・マネジャ916と、プロトコル・フィルタ918とが、ブロックの2つの組として図解され、出ていく送信グループと入ってくる受信グループとの関係でのそれぞれの役割が明確に示されている。

リクエストがOSM910から受信されると、IOPドライバ912は、次のリクエスト・メッセージ・フレームのための次の空のメッセージ・フレームのアドレスを取得する（ステップA1）。IOPドライバ912は、Free_List循環キューの中のHead_Pointerに記憶されている空のメッセージ・フレーム・アドレス（EMF_ADR）を検索することによって、これを行う。すると、Head_Pointerが、インクリメントされる。図10は、本発明の好適実施例によるFree_List循環キュー1000とPost_List循環キュー1002とを図解している。

IOPドライバ912は、検索された空のメッセージ・フレーム・アドレスにリクエスト・メッセージ・フレームを記憶する（ステップA2）。次に、IOPドライバ912は、メッセージ・フレーム・アドレスを、Post_List循環キューの中に位置するTail_Pointerの位置に記憶する。すると、Tail_Pointerは、インクリメントされる（ステップA3）。IOPドライバ912は、メッセージ・トランスポート・マネジャに、処理すべきリクエスト・メッセージ・フレームが存在することを告知する。このメカニズムは、レジスタ／割り込みベースの作用である。

次に、メッセージ・トランスポート・マネジャ914が、リクエスト・メッセージ・フレームのアドレスを、システム・メモリ902の中にあるPost_List循環キューの中のHead_Ponterから受け取る(A5)。メッセージ・トランスポート・マネジャは、1つ又は複数のリクエスト・メッセージ・フレーム・アドレスを、ローカル・メモリに記憶し、リクエスト・メッセージ・フレームを処理することができる。このオプションにより、いくつかのシステムのパフォーマンスを改善することができる。次に、メッセージ・トランスポート・マネジャ914は、リクエスト・メッセージ・フレーム・アドレスを受け取った後で、システム・メモリ902の中のPost_List循環キューにおいてHead_Ponterをイン

クリメントさせる(ステップA6)。その後で、メッセージ・トランスポート・マネジャ914は、インターフェース・マネジャに告知し、リクエスト・メッセージ・つふれーむ・アドレスを提供する(ステップA7)。それに応答して、インターフェース・マネジャ916は、リクエスト・メッセージ・フレームのアドレス及びサイズを用いて、IOPシステムDMA920をプログラムする(ステップA8)。

IOPシステムDMA920は、PCIバスのために仲裁を行い、リクエスト・メッセージ・フレームをローカル・メモリの中に移動させる(ステップA9)。インターフェース・マネジャ916は、IOPシステムDMA920に、1つ又は複数のメッセージ・フレームを、それがFree_List循環キューの中のTail_Ponterを更新する前に、ローカル・メモリの中に移動させることができる。このオプションは、いくつかのシステムのパフォーマンスを改善することができる。次に、インターフェース・マネジャ916は、メッセージ・トランスポート・マネジャ914に、1つ又は複数のリクエスト・メッセージ・フレームをローカル・メモリ904の中に移動させたことを告知する(ステップ10)。

メッセージ・トランスポート・マネジャ914は、新たなリクエスト・メッセージ・フレームのアドレスを、Free_List循環キューの中のTail_Ponterの位置に配置し、次に、Tail_Ponterをインクリメントする(ステップA11)。これを行うことによって、リクエスト・メッセージ・フレームは、空のメッセージ・フ

レーム・リソースに変換され戻される。結果的に、インターフェース・マネージャ 916は、プロトコル・フィルタ 918を助け、処理のためのリクエスト・メッセージ・フレームが入手可能であることを告知する(ステップ A12)。プロトコル・フィルタは、ローカル・メモリ 904からのそれぞれのリクエスト・メッセージ・フレームを検索して処理する(ステップ A13)。

プロトコル・フィルタ 918は、リクエスト・メッセージ・フレームの中の情報を用いて、1つ又は複数の送信コンテキスト・ブロック(TCB)を構築する(ステップ A14)。TCBは、ローカル・メモリ 922に記憶され、送信コンテキスト・マネージャ 924によって用いられる。いくつかのリクエスト・メッセージ・フレームは、エクスチェンジを構築するのに十分な情報を含んでいる。構築

するエクスチェンジを管理するのは、プロトコル・フィルタ 918の仕事である。これらのエクスチェンジは、単純なログイン・エクスチェンジ又はより複雑な SCSI/Oであり、コマンド、データ、Transfer_Rdy、応答シーケンスなどを含む。

送信コンテキスト・マネージャ 924は、使用可能になると常にTCBを送信する(ステップ A15)。送信コンテキスト・マネージャ 924は、どのようなエクスチェンジ情報も知っておらず、フレーム及びシーケンス・コンテキストだけを知っている。送信コンテキスト・マネージャ 924は、ローカル・メモリ 904から、最上位のTCBを選択し、フレーム 926に必要なコンテキスト情報を作成するだけでなく、必要な散乱/集合(S/G)エントリを送信S/G FIFOに提供する。送信コンテキスト・マネージャ 924は、S/Gエントリを送信S/G FIFO 928の中に配置し、コンテキスト情報をフレーム 926の中の配置する(ステップ A16)。また、ステップ A16において適切なときに、送信DMA(TX_DMA) 930が、S/G FIFO 928の最上位にあるS/Gエントリのアドレス及びサイズを用いてプログラムされ、PCIバスを求めて仲裁を行い、システム・メモリからデータを受け取り、それを送信バッファ 932の中に記憶する。

すべてのフレームの中のすべてのデータがTX_DMA経路から来るとは限らないことに注意することは重要である。例えば、ログイン・フレームに含まれる

116バイトのデータは、送信コンテキスト・マネジャ924を介してDMAバッファ932の中に配置される。送信コンテキスト・マネジャ924は、ローカル・メモリ904からログイン・データを検索することによってこれを行う。PCIバス上を移動することは全く必要ない。これは、明らかなパフォーマンス上の効果である。TX_DMA経路を用いない別の明らかなタイプのフレームは、リンク制御フレームである。リンク制御フレームの全体は、1つのTCBの中に含ませることが可能であり、送信コンテキスト・マネジャ924は、このTCBを受信すると、それを単純に、フレーム926に向けて経路決定する。フレーム(Framer)926は、このデータ及びコンテキスト情報を用いて、リンク・コントローラ934のための1つ又は複数のフレームを作成する。

リンク・コントローラ934は、2つのポートの間のリンクを管理する。例え

ば、フレームがリンクを介して送り出される準備ができていたときには、リンク・コントローラ934は、ループのための仲裁を行い(FC仲裁型ループ・トポロジを想定する)、仲裁を獲得すると、別の宛先であるNL_Portを開き、フレームをそこまで送る。

いったんTCBとそれに関連するフレームが送信されると、送信コンテキスト・マネジャ924は、プロトコル・フィルタ918に告知する(ステップA17)。プロトコル・フィルタ918は、TCBエントリを更新して、現在のTCBの完成を反映させる(ステップA18)。TCBエントリがリンクされたリストにおいてリンクされている場合には、プロトコル・フィルタ918は、リンクされたリストにおけるポインタを調整することによって、完成したTCBエントリを除去することができる。

リンク上を送信されているすべてのデータは、結果的には、別のポートにとっては受信データとなる。図9に図解されているポート936に入るデータは、最初に、ギガボー・リンク・モジュール(GLM)938を介して入る。このデータは、リンク・コントローラ934まで送られ、そこで、早期の宛先認識が生じる。入ってくるフレームがポート936に対するものである場合には、リンク・コントローラ934は、このフレームをフレーム・ディテクタ940に送る。フ

フレームがいったんフレーム・ディテクタ940によって検出されると、ヘッダが取り外され、コンテキスト情報がRXコンテキスト・マネージャ942によって作成される(ステップ19)。入ってくるフレームは、送信されたTCBに対する応答か、又は、未処理のフレームでありうる。フレームが送信されたTCBへの応答である場合には、コンテキストは、既に、プロトコル・フィルタによるエクスチェンジ管理の状態によって定義されている。システム・メモリに向けられた任意のデータは、受信バッファ(RX_Buffer)に入れられ、そのデータに対するS/GエントリがS/G FIFO928の中に配置される(ステップA20)。

フレームが未処理のフレームである場合には、コンテキストが発生される必要がある。最も単純なケースは、SCSI相互ロック・エクスチェンジの場合のように、フレームがコマンド情報を含む場合である。RXコンテキスト・マネージャ942は、ペイロードにおけるFCフレーム・ヘッダとSCSIコマンド

とから、必要な情報を作成する。ステップA20では、このフレームがデータも含む場合には、このデータは、RX_Bufferに配置され、S/Gエントリは、S/G FIFO928の中に入れられる。データがいったんRX_Buffer944の中にあり、S/GエントリがS/G FIFO928の中にあるときには、受信DMA(RX_DMA)946は、S/Gのアドレス及びサイズを用いてプログラムされる。RX_DMA946は、次に、PCIバスを求めて仲裁を行い、データを、このバスを介して、システム・メモリに転送する。

コンテキスト情報は、RXコンテキスト・マネージャ942からプロトコル・フィルタ918に送られ(ステップA21)、そこで、コンテキストに応じて複数の事柄が生じ得る。プロトコル・フィルタ918又はRXコンテキスト・マネージャ942がリンク応答フレームを送る必要があるときには、常に、これは、TCBエントリをTCBリンクされたリストの最上位に追加することによってなされる(ステップA22)。この例は、プロトコル・フィルタ918が1つ又は複数の受信フレームのためにACKフレームを発生する必要があるときである。

プロトコル・フィルタ918は、リクエストされたメッセージ・フレームが完成されたことを告知されると、OSMのためのリプライ・メッセージを作成し、

プロトコル・フィルタ918は、リプライ・メッセージをローカル・メモリ904の中に配置する(ステップA22b)。

コンテキスト及びプロトコル情報を作成するのに十分な情報を含む入来任意の未処理 (unsolicited) のフレームによって、プロトコル・フィルタ918は、受信コンテキスト・ブロック (RCB) が構築される(ステップA22c)。この例は、本発明のシステムがSCSIターゲット・モードにあるかどうかである。コマンド記述子ブロックを含むフレームが入ってくるときには、プロトコル・フィルタ918は、RCBリンクされたリストを発生し、ターゲットの視点からエクステンジ状態を管理することが必要となる。

プロトコル・フィルタ918は、いったん1つ又は複数のリプライ・メッセージ・フレームを作成すると、インターフェース・マネジャ916に、OSM910に送られる必要があるリプライ・メッセージ・フレームが存在していることを告知する(ステップA23)。インターフェース・マネジャ916は、次に、メッ

セージ・トランスポート・マネジャ914にリクエストを送り、リプライ・メッセージ・フレームのためのアドレスを取得する(ステップA24)。メッセージ・トランスポート・マネジャ914は、リプライ・メッセージ・フレームのための次の空のメッセージ・フレームのアドレスを取得する(ステップA25)。ステップA25では、メッセージ・トランスポート・マネジャ914は、システム・メモリにあるFree_List循環キュー内のHead_Pointerに記憶されている空のメッセージ・フレーム・アドレスを検索することによってこれを行う。次に、Head_Pointerがインクリメントされる。

メッセージ・トランスポート・マネジャ914は、この空のメッセージ・フレーム・アドレスと共に、インターフェース・マネジャ916を提供する(ステップA26)。インターフェース・マネジャ916は、リプライ・メッセージ・フレームの検索された空のフレーム・アドレス及び長さを用いてIOPシステムDMA920をプログラムする(ステップA27)。次に、IOPシステムDMA920は、PCIバスのための仲裁を行い、PCIバスを獲得したときには、システムDMA920は、リプライ・メッセージ・フレームをシステム・メモリ90

2の中にある空のメッセージ・フレーム・アドレスに移動させる(ステップA 2 8)。いったん1つ又は複数のリプライ・メッセージ・フレームが転送されると、インターフェース・マネジャ916は、メッセージ・トランスポート・マネジャ914に告知する(ステップA 2 9)。告知されると、メッセージ・トランスポート・マネジャ914は、リプライ・メッセージ・フレーム・アドレスを、Post_List循環キューの中のTail_Pointerの位置に記憶し、Tail_Pointerは、インクリメントされる(ステップA 3 0)。

メッセージ・トランスポート・マネジャ914は、IOPドライバ912に、処理するためのリプライ・メッセージ・フレームが存在することを告知する(ステップ31)。示されている例では、このメカニズムは、レジスタ/割り込みベースの作用である。IOPドライバ912は、Post_List循環キューの中のHead_Pointerから、リプライ・メッセージ・フレームのアドレスを検索する(ステップ32)。IOPドライバ912は、1つ又は複数のリクエスト・メッセージ・フレーム・アドレスを、システム・メモリの中に記憶することができ、そして、

リプライ・メッセージ・フレームを処理する。このオプションは、いくつかのシステムのパフォーマンスを改善せう。ステップA 3 2の場合のように、Post_List循環キューの中のHead_Pointerがインクリメントされる。IOPドライバ912は、リプライ・メッセージ・フレームを検索し、それらを、OSMに送り戻す(ステップA 3 2)。IOPドライバ912は、現在の新たなリプライ・メッセージ・フレームのアドレスを、Free_List循環キューの中のTail_Pointerの位置に配置し、Tail_Pointerをインクリメントする(ステップA 3 4)。こうすることによって、リプライ・メッセージ・フレームは、空のメッセージ・フレーム・リソースに変換して戻される。

送信コンテキスト・マネジャ877は、多数の責任を有している。これには、TCBの送信キューから、プロトコル・エンジン802によって作成されるデータを読み出すことが含まれる。TCBは、メモリ・データ経路844に接続されたローカル・メモリに位置している。送信コンテキスト・マネジャ877は、また、TCBが存在し、送信機858がアイドルであるときに、送信レジスタと、

バッファと、送信DMAユニット854（レジスタ又はFIFOレジスタ）とをロードする。更に、送信コンテキスト・マネージャ877は、S/Gリストからの必要に応じて、送信DMAユニット854にS/Gエントリを与え、プロトコル・エンジン802への割り込みを除去する。送信コンテキスト・マネージャ877は、また、それぞれのTCBが、ポインタをTCBデータ構造のリンクされたリストに再度書き込むことによって送信を終了したときに、TCBを、送信キューからフリー・キューへ再度リンクする。

次に図11を参照すると、本発明の好適実施例に従って送信コンテキスト・マネージャにおいて実現されているプロセスを図解する流れ図が、示されている。このプロセスは、IOPからのキックが生じたかどうかを判断することによって開始する（ステップ110）。「キック」は、Txコンテキスト及びレジスタ・ユニット868への権利（right）である。このプロセスは、IOPからのキックが生じるまで、ステップ1100に戻り続ける。この時点で、転送キュー・ヘッダ・ポインタがレジスタから読み出される（ステップ1102）。その後で、第1のTCBが読み出され（ステップ1104）、右フレーム・ヘッダが送信機に書き込ま

れる（ステップ1106）。次に、このフレームに対するペイロードがローカル・ペイロードであるかシステム・ペイロードであるかに関する判断がなされる（ステップ1108）。ペイロードがローカル・ペイロードである場合には、そのペイロードは、送信機バッファに書き込まれる（ステップ1110）。示されている例では、送信機バッファは、TxOバッファ867である。次に、シーケンスが送信されたかどうか判断される（ステップ1112）。シーケンスは、フレーム・ユニット865によって作成される一連のフレームであり、Txコンテキスト及びレジスタ・ユニット868におけるレジスタの単一のプログラミングによって開始される。プロセスは、シーケンスが送信されるまでは、ステップ1112に戻り続ける。シーケンスが送信されると、TCBが送信機キューから除去され（ステップ1114）、フリー・キュー・テール・ポインタが読み出される（ステップ1116）。この読み出しは、Txコンテキスト及びレジスタ・ユニット8

68からである。TCBは、フリー・キュー・テールにリンクされる(ステップ1118)。次に、フリー・キュー・テール・ポインタが更新され(ステップ1120)、送信キュー・ヘッダ・ポインタが更新される(ステップ1122)。ポインタの更新は、Txコンテキスト及びレジスタ・ユニット868に対して右側である。そして、TCBが存在するかどうかの判断がなされる(ステップ1124)。別のTCBが存在する場合には、プロセスは次のTCBを読み出し(ステップ1126)、プロセスは、既に述べたように、ステップ1106に進む。別のTCBが存在しない場合には、キューの最後に到達し、プロセスは、ステップ1100に戻る。

再びステップ1108を参照すると、ペイロードがシステム・ペイロードである場合には、プロセスは、S/GエントリをS/Gリストから読み出す(ステップ1128)。次に、S/Gエントリが、送信S/G FIFOの中にロードされる(ステップ1130)。そして、S/Gエントリが更に存在するかどうか判断される(ステップ1132)。これ以上S/Gエントリが存在しない場合には、プロセスは、既に述べたように、ステップ1112に進む。それ以外の場合には、プロセスは、S/G FIFOがいっぱいであるかどうかを判断する(ステップ1134)。S/G FIFOがいっぱいではない場合には、プロセスは、ステップ1128に戻る。そうでなければ、プロセスは、FIFOがいっぱいでなくなるまで、

ステップ1134に戻り続ける。

受信コンテキスト・マネージャ878は、ファイバ・チャネルの受信コンテキスト管理に自動を提供し、それによって、他のデバイス及び／又はプロトコル・エンジン802などのシステム・リソースのワークロードを軽減する。受信コンテキスト・マネージャ878は、コンテキスト管理、DMA開始、DMA更新、フレーム完成(complete)処理など、様々な機能を提供する。コンテキスト管理には、ファイバ・チャネルのヘッダ情報を受信制御ブロック(RCB)に調停(reconcile)することを含むが、これは、ファイバ・チャネル・シーケンス情報を有効化しデータ転送パラメータを特定する手段を提供する。DMA開始機能は、フ

ファイバ・チャンネル・ヘッダ・パラメータ・フィールドを正しいバッファ・オフセットにマップしDMA転送を開始することによって、初期開始点を用いて受信DMA 856をプログラムすることを含む。DMA更新機能には、フレームDMA転送を維持するのに要求される追加的なバッファ・アドレス/長情報を用いて、受信DMA 856を更新することが含まれる。フレーム完成処理では、受信DMA情報の更新と、ファイバ・チャンネル・シーケンス完成の検出と、条件的な完成報告とが、生じる。

次に、図12を参照すると、本発明の好適実施例による受信された制御ブロックのためのフォーマットが示されている。受信制御ブロック1200は、ファイバ・チャンネル・シーケンスを管理するのに要求される情報を含む。ファイバ・チャンネル・ヘッダ・フィールド1202は、入来フレームを有効化するのに用いられる情報を含む。シーケンス状態情報フィールド1204は、ファイバ・チャンネル・シーケンスをトラッキングし管理するのに用いられる。DMA情報フィールド1206は、ファイバ・チャンネル・データの宛先アドレスへのマップをトラッキングし管理するのに用いられる。最後に、タイム・スタンプ・フィールド1208は、シーケンスが完成したことを指示するのに用いられる。

次に、図13を参照すると、本発明の好適実施例によるコンテキストのスイッチングを実行するのに用いられる流れ図が示されている。コンテキストのスイッチングは、フレーム受信機が受信されたコンテキスト・マネージャにコンテキストのスイッチングが必要であることを告知するときに開始する。この状況は、現在

のファイバ・チャンネル・ヘッダが現在確立している受信コンテキストと一致しないときに生じる。この時点で、受信コンテキスト・マネージャは、RCBの中にある次のコンテキストを見出す(ステップ1300)。次に、そのフレームがこのRCBに対する新たなファイバ・チャンネル・シーケンスの第1のフレームであるかどうかに関する判断がなされる(ステップ1302)。受信されたフレームが特定のシーケンスに対して受信された第1のフレームであるかどうかという判断は、RCBのコンテキスト状態ワードの中の「アクティブ」ビットに基づく。このビットは、当初、プロトコル・エンジンによって0に設定され、この特定のシーケ

ンスがアクティブではない(フレームがまだ受信されていない)ことを指示する。受信コンテキスト・マネージャは、コンテキストのルックアップ及びスイッチングを実行し、関連するRCBがまだこのビット・セットを有していないと判断するときには、ステップ1304に説明されている作用を実行し、RCBのアクティブ・ビットを1に設定する。回答が肯定である場合には、RCBは、シーケンスID(S_ID)と受信ID(RX_ID)とを用いて更新され、シーケンス・カウント(SEQ_CNT)を用いて、このシーケンスに対して予測される値が設定される(ステップ1304)。

次に、フレーム受信コンテキストが有効であるかどうか判断される(ステップ1306)。フレーム受信コンテキストは、先行するシーケンスが不完全である場合には、典型的には有効である。ステップ1306は、先行する完全なシーケンスの状態を記憶し、それによって、シーケンスは、別のフレームがそのシーケンスに対して受信されるときには後の時点で完成されることになる。この判断は、また、第1のフレームがこのRCB出ない場合には、ステップ1302から直接になされる。フレーム受信コンテキストが有効である場合には、先のフレーム受信コンテキストは、メモリにセーブされ(ステップ1308)、新たなフレーム受信コンテキストがメモリからロードされる(ステップ1310)。フレーム受信コンテキストが有効でない場合には、プロセスはステップ1308をスキップして、メモリからの新たなフレーム受信コンテキストをロードするステップ1310に進む。次に、信号が受信機に送られ、フレーム受信コンテキストを再評価する(ステップ1312)。プロセスは、その後で終了する。

次に、図14を参照すると、本発明の好適実施例によるDMA開始プロセスの流れ図が示されている。フレーム受信機がファイバ・チャンネル・ヘッダを評価し、このヘッダが先にロードされ現在確立している受信コンテキストと一致していると判断するときには、フレーム受信機は、受信コンテキスト・マネージャに、フレーム転送が必要であることを告知する。この時点で、受信コンテキスト・マネージャは、DMA開始プロセスを開始させる。このプロセスは、フレーム相対オフセットがRCBの現在相対オフセットよりも大きいかどうかを判断することによ

って開始する(ステップ1400)。この判断への回答が否定であるときには、プロセスは、正しいエントリが見つかるまでS/Gリストを逆方向に走査する(ステップ1402)。回答が肯定である場合には、正しいエントリが見つかるまでS/Gリストを順方向に走査する(ステップ1404)。ステップ1400ないし1404は、ファイバ・チャネルのヘッダ・パラメータ・フィールドをRCBの現在の相対オフセット・フィールドと比較し、RCBのベースS/GポイントとRCBの現在のS/Gポイントとを用いてDMA S/Gリストを順方向又は逆方向に走査し、正しい開始S/G要素を見つけることによって、適切な開始DMA S/G要素(RCB現在のS/Gポイント)を見つけるのに用いられる。これらのステップにより、順不同(out of order)で受信されたフレームが正しいDMA バッファ・アドレスに適切にマップされることが可能となる。

次に、アドレス及び長さの情報が調整される(ステップ1406)。このステップは、見出されたS/G要素内のこのフレームに対する実際の開始及び長さを、ファイバ・チャネルのヘッダ・パラメータ・フィールド及びRCBの現在のS/Gポイント・アドレスと、RTCBの現在のS/Gポイントの長さ、RCBの現在の相対オフセットとに基づいて、計算する。次に、受信DMAは、アドレス及び長さ情報を用いてプログラムされ、DMA転送が開始される(ステップ1408)。プロセスは、その後で終了する。

次に図15を参照すると、本発明の好適実施例によるDMA更新プロセスの流れ図が示されている。ファイバ・チャネルのフレーム・ペイロード・データが複数のDMA S/Gエントリにわたるときには、受信コンテキスト・マネージャは、追加的なDMAプログラミング情報を受信DMAユニットに提供し、それによ

て、フレームDMA転送が継続することになる。受信DMAユニットは、追加的なS/Gエントリの必要性を告知するが、これにより、次のS/G要素を取得しこのS/G要素に関係する現在の相対オフセットを更新することによって(ステップ1500)、プロセスが開始する。その後で、受信DMAユニットは、ステップ1500において決定されたアドレス及び長さ情報を用いてプログラムされ、DMA転送は、データの転送を継続を開始する(ステップ1502)。プロセス

は、この後で終了する。

次に、図16を参照すると、本発明の好適実施例によるフレーム完成処理プロセスが示されている。受信DMAユニットからの信号のフレーム転送によって指示されているように、受信DMA動作の完了の際に、受信コンテキスト・マネージャは、RCBにおけるDMA転送情報を更新することによって、フレーム完成処理を開始する(ステップ1600)。その後で、シーケンスの最後に到達したかどうかに関する判断がなされる(ステップ1602)。シーケンスの最後に到達した場合には、タイマが読み出され、RCBタイム・スタンプ・フィールドが、EMBレジスタ及びタイマ・ユニット850からタイマ値と共に書き込まれる(ステップ1604)。その後で、完成報告が禁止されているかどうか判断される(ステップ1606)。報告が禁止されていない場合には、RCBポインタが、シーケンス完成キューに書き込まれる(ステップ1608)。プロセスは、その後で終了する。

再びステップ1606を参照すると、報告が禁止されるべき場合には、プロセスは、終了する。プロセスは、また、ステップ1602においてシーケンスの最後に到達した場合にも終了する。

リンク制御ユニット864は、2つのポートの間のリンクである。リンク制御ユニット864は、チップ800内でデータの送受信をする際に用いられる。リンク制御ユニット864によって提供される機能は、リンクの幾何学的態様(トポロジ)に依存する。例えば、リンクがポイント間のものである場合には、リンク制御ユニット864は、データの転送を提供するだけである。仲裁されたループ・トポロジが用いられている場合には、リンク制御ユニット864は、ループの管理する追加的な機能を提供する。「ループ」という用語は、一方向的にデータ

を転送するように接続された集合ノードを意味する。それぞれのノードは、ループ上のデータに対するソース又は宛先であり、例えば、アダプタ、コンピュータ、遠隔ストレージ・ユニットなどでありうる。

次に図17を参照すると、本発明の好適実施例によるループ管理制御のための

ステート・マシンが示されている。転送制御ユニット880の内部に実現されるステート・マシン1700は、リンク制御ユニット864である。リンク制御ユニット864は、示されている例では、仲裁されたループへのアクセスを提供している。仲裁されたループでは、仲裁プロセスは、どのノードが送信データへの権利を有しているのかを判断するのに用いられる。リンク制御ユニット880は、ループに対するリクエストを送出することによってデータ転送のための仲裁されたループへのアクセスを得ようと試みる。リクエストは、ARBプリミティブとも称される仲裁プリミティブを送ることによってなされる。ループが取得されると、本発明は、ノード及びループのアクティビティをモニタして、ループの全体的なパフォーマンスを最大化する。本発明は、トラフィックがすぐに準備が完了するのか、停止するのかを識別して評価し、更に、ループのパフォーマンスを最大化する際にループ上でどのようなアクティビティが要求されているのかを識別し、評価する。

ステート・マシン1700は、アイドル状態であるステートS1で始まり、特定の宛先亜又はノードに対してループを取得するというリクエストがなされるまで、この状態にとどまる。ループが取得されるべきときには、状態S1にあるステート・マシン1700は、ループに対し、仲裁プロセスに入ってループの所有権を取得するように求める。ループは、データがターゲット又は宛先ノードへの送信のために読み出されるときにだけ、又は、宛先ノードへ送信されるデータが入手可能であることがわかるとすぐにだけ、要求される。

ループを取得することを求めるリクエストに応答して、ステート・マシン1700は、ループ状態を待機する状態S2にシフトし、そこで、ステート・マシン1700はループの所有権が得られた後で開放(open)プリミティブを送出する。状態S2では、ステート・マシン1700は、ループが、ステート・マシン1700が実行しているノードによるデータ転送のために使用可能となるのを待機す

る。ノードは、ループを求めて仲裁を行いループを獲得したときには、仲裁獲得状態にある。「開放」(OPEN)プリミティブは、そのプリミティブがターゲット

・ノードを特定するループの上に送られる。ノードは、OPNプリミティブが送られるときには、「開放」ノードであると考えられる。開放ノードは、ソース・ノードである。開放ノードは、ループを求める仲裁プロセスに再度入ることを要求されることなく、異なるノードへの接続を断絶させたり確立したりする。「開放された」ノードは、開放ノードが接続を確立しているノードである。開放されたノードは、宛先又はターゲット・ノードである。開放されたノードは、どこかが入手可能であるならば、データを開放ノードに戻すことができるが、他のどのノードにもデータを送ることは許されない。「接続」は、ソース・ノードがデータを送ることを希望している宛先ノードを識別する開放ノードによってなされる。接続は、OPNプリミティブが送られるときに確立される。接続を確立するには、完全なハンドシェークは要求されない。接続の閉鎖だけが、完全なハンドシェーク、すなわち、プリミティブの変換を必要とする。

ステート・マシン1700は、接続が要求されたノードに対し確立されることに応答して、オン・ループ状態である状態S3にシフトし、規則によって、フレームの送出が可能となる。「フレーム」とは、ヘッダ情報が追加されたデータの packets である。一般に、すべてのデータがオン・デマンドで入手可能であることがわかるまではフレームを送り始めることは不可能である。その理由は、フレームの送信は、示されている例では、中断することができないからである。ステート・マシンが状態S3にあるときには、フレームは、ループ上をノードまで送ることが可能である。フレームの完成に応答して、ステート・マシン1700は、判断ウィンドウ状態である状態S4にシフトして、追加的なフレームが送る準備ができているかどうかを検討される。また、追加的なフレームが先のフレームと同じノードに送られる準備ができており、更に、宛先が追加的なフレームを受け入れる準備ができている場合には、ステート・マシン1700は、状態S3に戻る。ハンドシェーキングのプロセスが、送信側のノードと受信側のノードとの間で生じ、受信側ノードにおけるバッファがオーバーランすることを防止する。このハンドシェーキングはまた、「クレジット」とも称される。状態S3と状態S4

との間のこのシフトは、データのフレームが宛先に向けて転送される準備ができている限りなされる。状態S4では、ノードに送るための追加的なデータが入手可能ではない、又は、規則がループの開放を命じている場合には、ステート・マシン1700は、閉鎖状態の待機である状態S5にシフトし、それ以上のデータはノードに送られない。この状態では、「閉鎖」(CLS)プリミティブがループ上に送られ、ループを開放する。閉鎖プリミティブのハンドシェーキングが完了しループがもはや開放ではなくなると、ステート・マシン1700は、状態S1のアイドル状態に戻る。閉鎖が生じると、ループの所有権 (ownership) は必ずしも放棄されるとは限らない。変化が、decision_window-waiting_for_close_idleから生じると、ループの所有権は放棄される。所有権の放棄を望まない場合には、decision_window-waiting_for_transfer-waiting_for_openループが選択される。これは、基本的には、先のルートと同じCLSハンドシェーキングを行うが、ループの所有権は保持される。これは、OPNをwaiting_for_open状態では直ちに送ることが許されないことの理由の一部である。

再び状態S4を参照すると、データの追加的なフレームが入手可能であり許容されているが、異なるノードに対するものであって異なるノードが開放されることを要求している場合には、ステート・マシン1700は、転送状態の待機である状態S6にシフトする。状態S6では、ステート・マシン1700は、別のノードへの転送リクエストを送る。状態S6におけるこのリクエストは、閉鎖プリミティブの送出と、戻されるべき閉鎖プリミティブの待機とを含む。先の接続が閉鎖されたと判断するが、規則によって開放を送ることが許されない場合には、ステート・マシン1700は、開放状態の待機である状態S7にシフトし、この状態において、ステート・マシン1700は、開放プリミティブを送り、新たなノードを開放する。状態S7では、ステート・マシン1700は、データ転送の前にタイム・ギャップが生じるように、待機する。要求されたノードへの接続が確立されると、ステート・マシン1700は、状態S3にシフトして、データフレームをそのノードに送る。

状態S6に戻ると、閉鎖プリミティブへの応答を待機している間に、ステート

マシン1700は、新たなノードが廃棄されるようにとのリクエストを要求する規則にตอบสนองして、状態S5にシフトし、ループを閉鎖する。状態S1では、閉鎖プリミティブが受信され、閉鎖プリミティブを伴う応答が要求されると、ステート・マシンは状態S5にシフトして閉鎖プリミティブを送り、状態S1に戻る。再び状態S2を参照すると、ループへのアクセスを待機している間に、ステート・マシン1700は、応答における閉鎖プリミティブを求める閉鎖プリミティブを受信することにตอบสนองして、又は、送信リクエストが廃棄されることを規則が求めている場合には、状態S5にシフトする。

次に図18を参照すると、本発明の好適実施例による開放状態におけるループ管理のためのプロセスの流れ図が示されている。このプロセスは、あるノードが、ステート・マシンが動作しているノードを開放したかどうかと、クレジットの不足のためにデータ送信が行われていないかどうかと、合理的な時間周期の後で接続を閉鎖していないかどうか(ステップ1800)を判断することによって開始する。データ転送がなされないままで選択された時間周期が経過した場合には、プロセスは、接続を閉鎖し(ステップ1802)、その後でプロセスが終了する。そのような状況が存在しない場合には、データ送信のリクエストが除去されたかどうかに関する判断がなされる(ステップ1804)。リクエストは、様々な理由で除去される。例えば、ホストからのリクエストや、エラー条件の回復、また、送信機がデータ送信のリクエストを除去することもありうる。リクエストが除去されると、接続は閉鎖され(ステップ1806)、ループの所有権は開放され(ステップ1808)、プロセスは、その後で終了する。

次の図19を参照すると、本発明の好適実施例によって、アイドル状態でループの取得を制御するのに用いられる規則を組み入れているプロセスの流れ図が示されている。示されている例では、ポートは「欲張りな」(greedy)状態におかれており、この状態では、いったん得られたループの所有権は、データが再び入手可能となる可能性が存在する限り、保持される。更に、この状態では、ノードは、別のノードがループへのアクセスを望んでいることを検出しない場合には、ループの所有権を保持する。

プロセスは、欲張りな状態が存在するかどうかを判断することによって開始す

る(ステップ1900)。欲張りな状態が存在する場合には、プロセスは、ノードがデータをロードしているかどうかを判断する(ステップ1902)。このノードは、ソース・ノードである。ノードがデータをロードしていない場合には、プロセスは、ステート・マシンをアイドル状態に維持する(ステップ1904)。プロセスは、その後を終了する。ノードがデータをロードしている場合には、ループが取得され(ステップ1906)、ステート・マシンが、図17に図解されているように、ループ待機状態にシフトされる(ステップ1908)。プロセスは、その後で終了する。再びステップ1900を参照すると、欲張りの状態が存在していない場合には、プロセスは、完全なフレームが送られる準備ができているかどうかを判断する。完全なフレームが送られる準備ができている場合には、プロセスは、ライブ(live)を取得するように進み(ステップ1906)、ループ待機状態にシフトされる(ステップ1908)。プロセスは、その後で終了する。完全なフレームを送る準備ができていない場合には、ステート・マシンは、アイドル状態に維持され(ステップ1912)、プロセスは、後で終了する。図19は、複数のステップがアイドル状態の間に反復される単一のパスを通過するプロセスを示している。

図20を参照すると、本発明の好適実施例に従ってループ待機状態において用いられる規則を組み入れているプロセスの流れ図が図解されている。このプロセスは、ノードがデータ転送のための別のノードによって開放された場合に用いられる。プロセスは、ループを取得しようとし(ステップ2000)、他方で、データを遠隔ノードに送ることを試み、このプロセスが実行されるノードを開放する(ステップ2002)。ループが取得されたかどうかに関する判断がなされる(ステップ2004)。ノードがまだ取得されていない場合には、プロセスは、ステップ2000に戻る。ループ取得の際には、プロセスは、ステップ2002においてデータを送るパラレル・プロセスを含めて、終了する。また、遠隔ノードに送るために追加的データが存在するかどうかを判断する(ステップ2006)。データが依然として入手可能である場合には、プロセスは、ステップ2002に戻る。送られるべき追加的データが存在しない場合には、プロセスは、ステップ2000におけるループ取得の試みの終了も含めて、終了する。このプロセスは、

デー

タ転送のために別のノードによってそのノードが開放されるかどうかを判断する。ただし、この遠隔ノードがループが取得されたときに転送されることが予定されているデータののための宛先ノードである。ノードが宛先ノードでもある別のノードによって開放された場合には、データは、好ましくは、この遠隔ノードに送られる。そして、更に多くのデータが送られるために存在しているかどうかを判断される(2004)。送られるデータが更に存在するならば、ステート・マシンは、ループを取得する(ステップ2008)。プロセスは、その後で終了する。再びステップ2004を参照すると、送るべきデータがそれ以上存在しない場合には、ステート・マシンは、閉鎖待機状態にシフトされる(ステップ2008)。プロセスは、その後で終了する。再びステップ2000を参照すると、ノードが遠隔ノードによって開放されていない場合には、プロセスは、ステップ2006においてループを取得するように進む。これらのステップはシリアルであるように示されているが、プロセスは、実際には、2つのパラレル・プロセスとして生じる。連続的にループを取得する間に、データは、可能であれば、送られるのが好ましい。これらのプロセスは、すべてのデータが送られるまで、又は、ループが要求されるまで、パラレルに継続する。

次に図21では、本発明の好適実施例に従って判断ウィンドウ状態における変化(transitions)を扱うのに用いられる規則を組み入れているプロセスの流れ図が示されている。プロセスは、ループに対するリクエストが別のノードによってなされたかどうかを判断することによって開始する(ステップ2100)。ループが別のノードによって要求されていない場合には、クレジットの不足のためにデータが送信されないかどうか、更に、データが受信されていないかどうかを判断される(ステップ2102)。このような条件が存在しない場合には、ループが保持されていた時間の長さをトラッキングするタイマが、リセットされる(ステップ2104)。プロセスは、その後で終了する。ステップ2102を参照すると、データが送信又は受信されていない場合には、プロセスは、ループが保持されていた時間が選択された時間よりも長いかどうかを判断する(ステップ21

08)。この選択された時間周期は、実現例に依存してプログラム可能である。
ループが保持されていた時間が選択された時間周期よりも長い場合には、ステー

ト・マシンは、閉鎖待機状態にシフトされ(ステップ2108)、プロセスは、その後で終了する。そうでない場合にも、プロセスは、停止する。ステップ2100を再び参照すると、ループに対するリクエストが別のノードによってなされ、プロセスは、また、既に述べたように、ステップ2106に進む。

次に図22を参照すると、本発明の好適実施例に従って判断ウィンドウ状態における変化を処理する規則を組み入れているプロセスの流れ図が示されている。このプロセスは、ノードが公平(フェア、fair)な状態である場合に実行される。換言すると、ノードは、残りのノードがループを要求するときには常にそれらの残りのノードがループにアクセスすることを可能にすることを試みる。このプロセスは、ループがリクエストされているかどうかを判断することによって開始する。そうでない場合には、ステート・マシンは、判断ウィンドウ状態から閉鎖状態にシフトして、接続を閉鎖し、ループを開放する(ステップ2202)。その後で、ステート・マシンは、アイドル状態においてループを要求することを試みるように促される(ステップ2204)。プロセスは、その後で終了する。

図23を参照すると、本発明の好適実施例に従って判断ウィンドウ状態からの変化を扱う際に用いる規則を組み入れたプロセスの流れ図が示されている。このプロセスは、ループ上のノードをポーリングする(poll)のに用いられる。このプロセスは、ノードを開放することによって開始する(ステップ2300)。次に、プロセスは、ノードが応答する時間周期の待機する(ステップ2302)。次に、追加的なフレームがどのようなものであれ受信されたかどうか判断される(ステップ2304)。追加的なフレームが受信されている場合には、プロセスは、ステップ2302に戻る。そうでない場合には、ノードは、閉鎖される(ステップ2306)。次に、追加的なノードがポーリングのために存在するかどうか判断される(ステップ2308)。追加的なノードが存在する場合には、プロセスは、ステップ2300に戻り、別のノードをポーリングする。そうでない場合には、プロセスは、終了する。

以上において、本発明は、完全に機能するデータ処理システムのコンテキストに関して説明してきたが、この分野の当業者であれば、本発明のプロセスは、命令のコンピュータ可読な媒体の形式や様々な形態で提供することが可能であり、

本発明は、提供されたものを実行するのに実際に用いられる媒体を生じる特定のタイプの信号とは無関係に応用されうることが重要である。コンピュータ可読な舞いたいには、フロッピー・ディスク、ハードディスク・ドライブ、RAM、CD-ROM、出る及びアナログ通信リンクなどの伝送タイプの媒体などが含まれる。

本発明の説明は、例示と説明を目的としてなされたが、網羅的なものではなく、ここで開示された形式に限定されるものでもない。当業者には、多くの修正や改変が明らかである。実施例は、当業者が考慮されている特定の使用に対して適した様々な修正を含んだ形で本発明を理解することを可能ならしめるように、本発明の原理を最良に説明するように選択され説明されてる。

【図1】

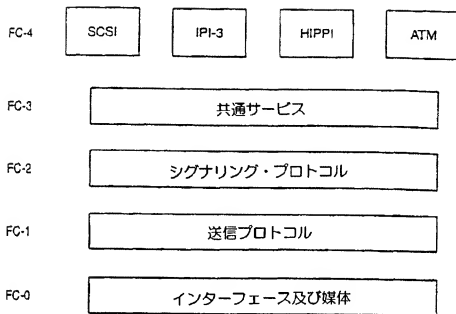


FIG. 1

【図2】

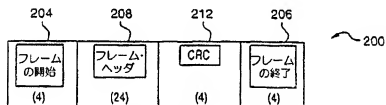


FIG. 2A

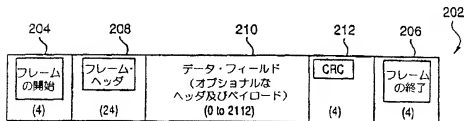


FIG. 2B

【図3】

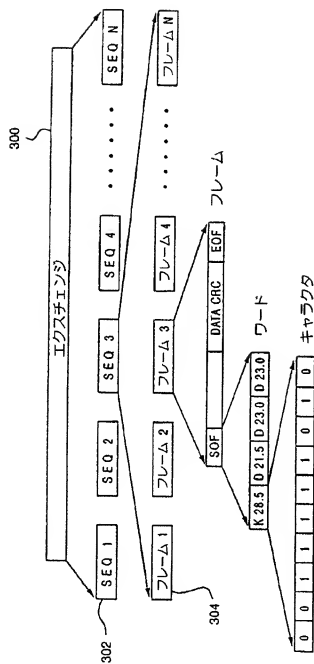


FIG. 3

【図4】

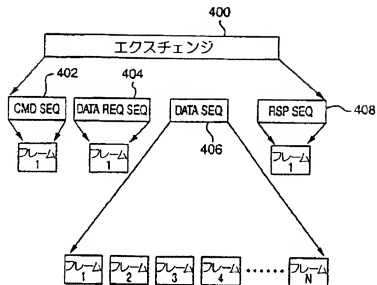


FIG. 4

【図5】

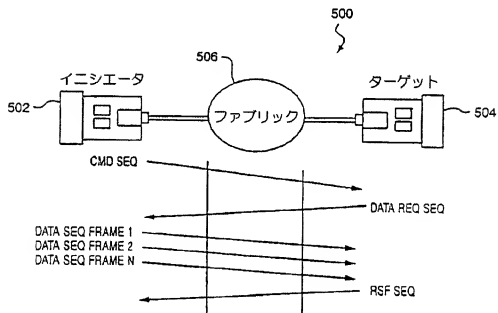


FIG. 5

【図6】

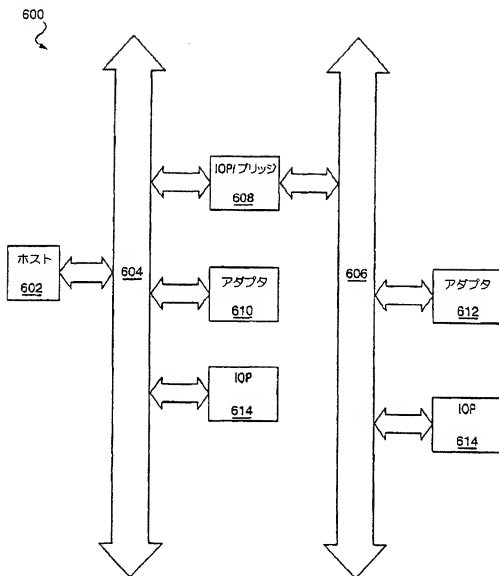


FIG. 6

【図7】

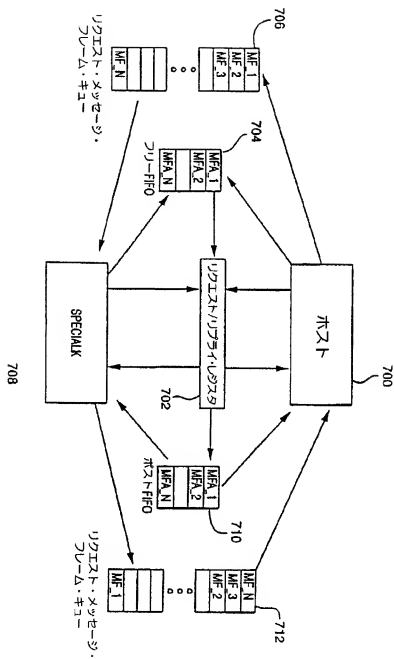


FIG. 7

【図8A】

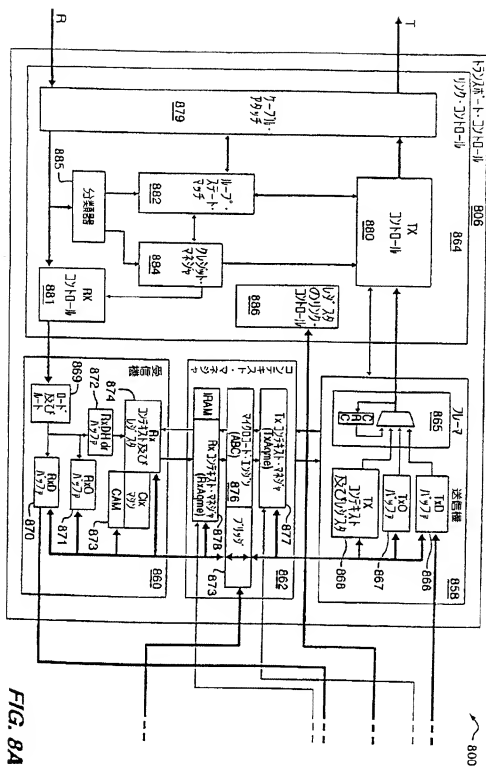
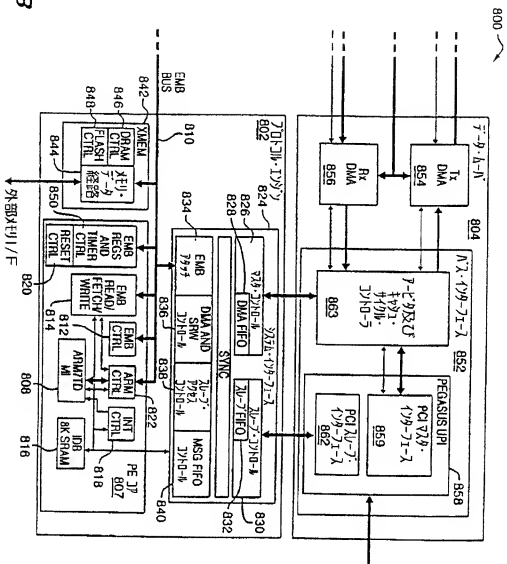


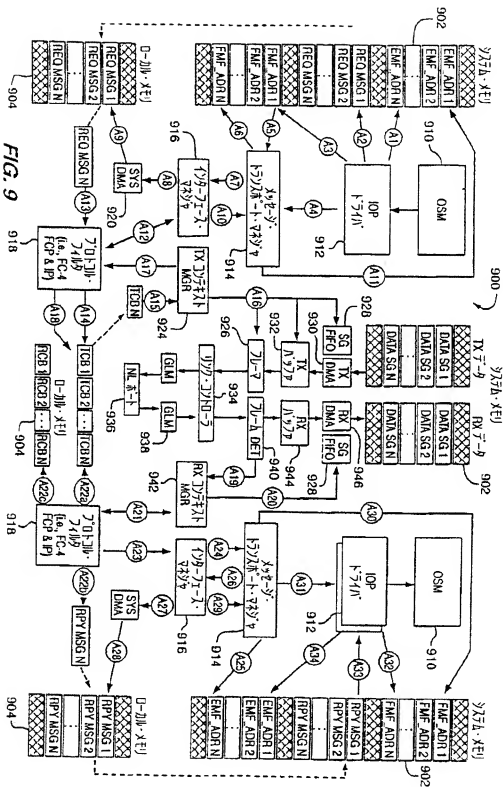
FIG. 8A

【図8B】

FIG. 8B



【图 9】



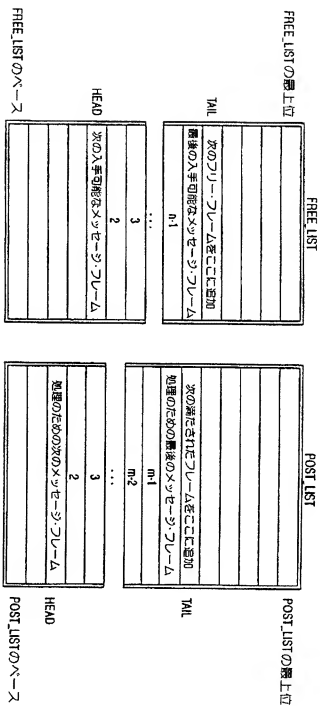
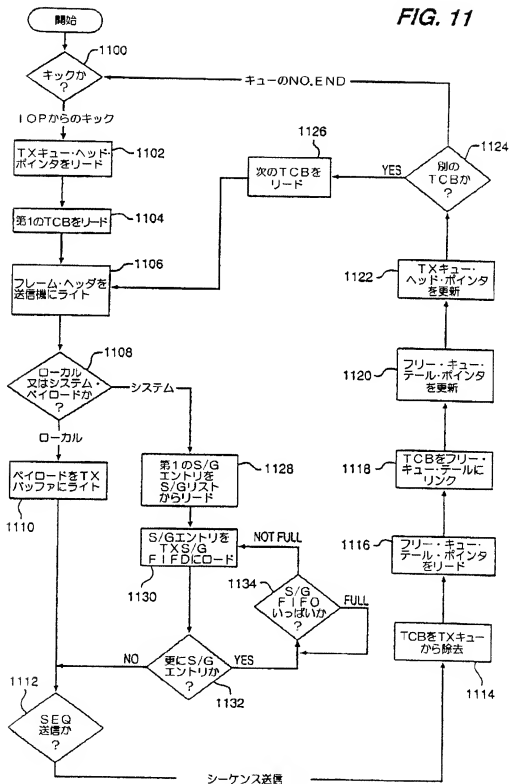


FIG. 10

【図10】

【図11】

FIG. 11



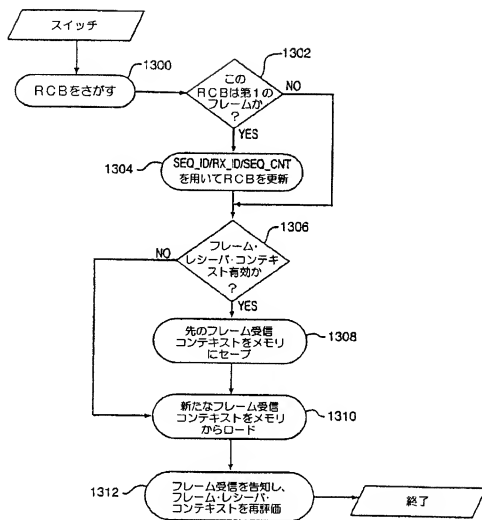
【図12】

FIG. 12

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----|-----|----------------|--|--|--|--|---------------|--|--|--|--|---------|--|--|--|--|-------|--|--|--|--|------|--|--|--|--|--|--|--|--|--|------|--|--|--|--|--|--|--|--|--|
| | | 1200 | | | | | | | | | | 1204 | | | | | | | | | | 1206 | | | | | | | | | | 1208 | | | | | | | | | |
| バイト | ワード | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | 0 | R_CTL | | | | | D_ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 1 | 予約済 | | | | | S_ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | 2 | タイプ | | | | | F_CTL | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| C | 3 | SEQ_ID | | | | | OF_CTL | | | | | SEQ_CNT | | | | | RX_ID | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 | 4 | OX_ID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 14 | 5 | 最高シーケンス・カウント | | | | | 最低シーケンス・カウント | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 18 | 6 | フレーム・カウント | | | | | 最後のシーケンス・カウント | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1C | 7 | シーケンス・バイト・カウント | | | | | コンテキスト・ステータス | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 20 | 8 | | | | | | ベース・S/Gポインタ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 24 | 9 | | | | | | 類似フレーム相対オフセット | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 28 | A | | | | | | 現在のS/Gポインタ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2C | B | | | | | | 現在の相対オフセット | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 30 | C | | | | | | タイム・スタンプ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 34 | D | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

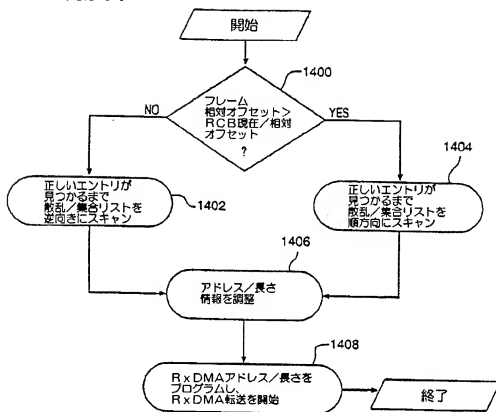
【図13】

FIG. 13



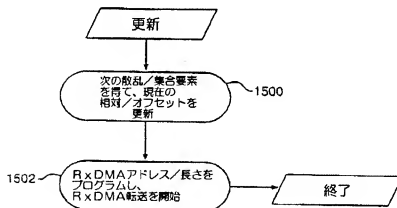
【図14】

FIG. 14



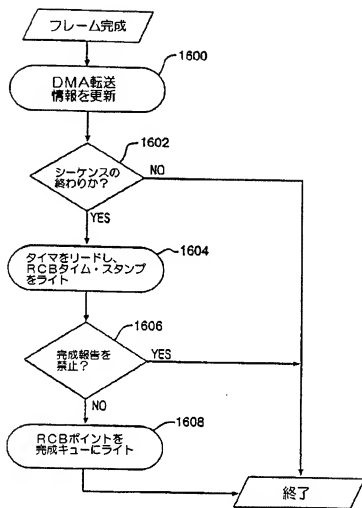
【図15】

FIG. 15



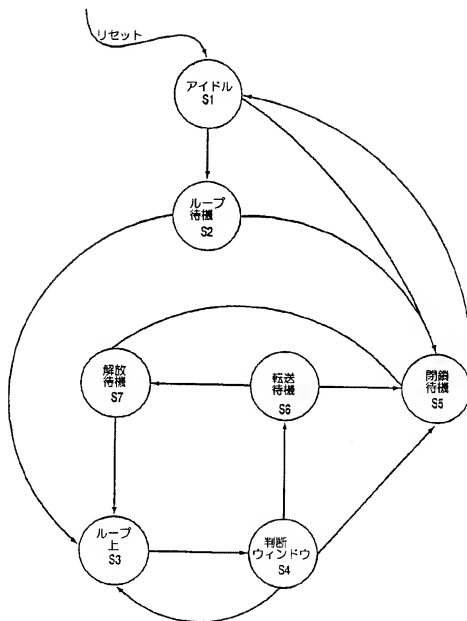
【図16】

FIG. 16



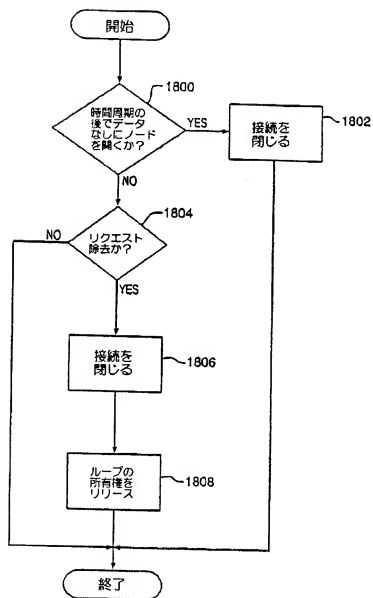
【図17】

FIG. 17



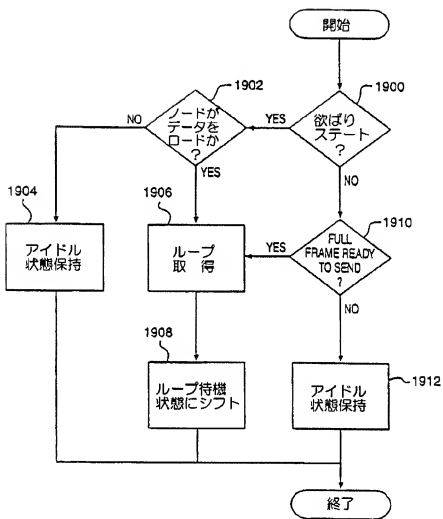
【図18】

FIG. 18



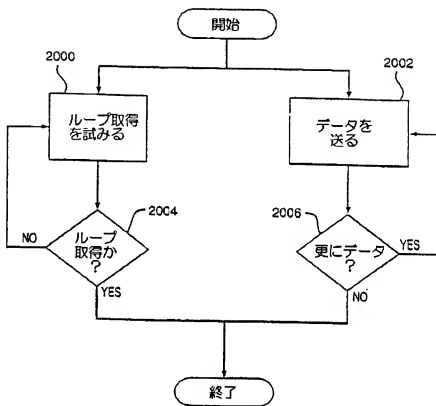
【図19】

FIG. 19



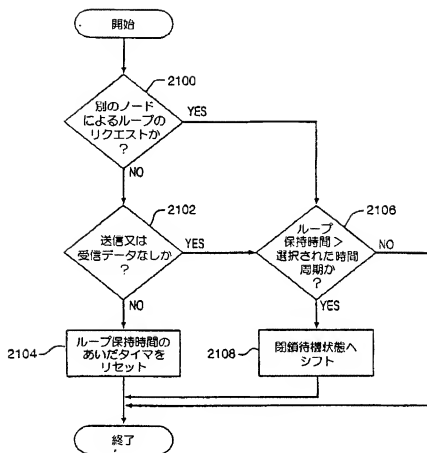
【図20】

FIG. 20



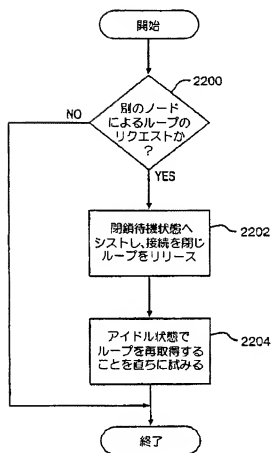
【図21】

FIG. 21



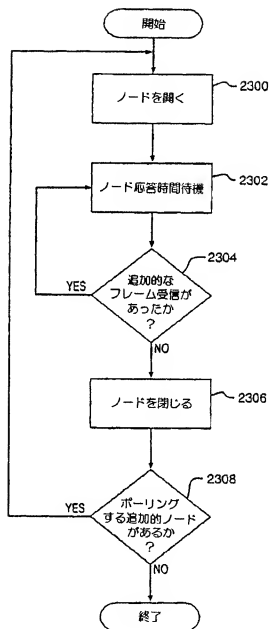
【図22】

FIG. 22



【図23】

FIG. 23



【手続補正書】特許法第184条の4第4項

【提出日】平成11年10月14日（1999. 10. 14）

【補正内容】

〔請求の範囲〕の記載を次の通りに補正する。

『1. チップであって、

入力ポートと、

出力ポートと、

第1の処理ユニットであって、

このチップから離れて位置するホストからのリクエストを受信し、データを宛先に送信する第1の受信手段と、

前記データを受信する第2の受信手段と、

前記宛先と前記データとを、前記宛先への送信に先だって、このチップに接続されたメモリに記憶する記憶手段と、

を含む第1の処理ユニットと、

第2の処理ユニットであって、

宛先に送信するデータの存在を検出する検出手段と、

前記データと前記宛先とを用いて前記宛先への搬送のためのフォーマットに前記データをフォーマットするフォーマット手段と、

を含む第2の処理ユニットと、

を備えていることを特徴とするチップ。

2. 請求項1記載のチップにおいて、前記第1の処理ユニットは、宛先に送られるデータのためのアクティビティのリストを作成し、前記検出手段は、前記リストを検査し前記宛先に送信するためのデータの存在を検出することを特徴とするチップ。

3. 請求項1記載のチップにおいて、予測されるデータのリストが前記第1の処理ユニットによって発生され、前記第2の処理ユニットは、

データを遠隔ソースから受信する受信手段と、

前記データを記憶する識別手段と、

前記第1の処理ユニットに、いつすべてのデータが前記遠隔ソースから受信さ

れたかを指示する指示手段と、

を含むことを特徴とするチップ。

4. 請求項1記載のチップ・アーキテクチャにおいて、前記第2の処理ユ

ニットは埋め込み型プロセッサであることを特徴とするチップ・アーキテクチャ。

5. 請求項1記載のチップにおいて、前記第2の処理ユニットはステート・マシンであることを特徴とするチップ。

6. 請求項1記載のチップにおいて、前記第2の受信手段は、ホスト・メモリからこのチップに結合されたメモリにデータを転送する転送手段を備えていることを特徴とするチップ。

7. 請求項1記載のチップにおいて、前記入力ポートは、ホスト・システム上のバスと通信するように構成され、他方で、前記出力ポートは、前記宛先と通信するように構成されていることを特徴とするチップ。

8. ホストから宛先にデータを転送するチップであって、
前記ホストから、前記宛先へのデータ転送のリクエストを受信し、メモリに前記宛先と前記データとのIDを記憶する第1の処理ユニットと、

前記メモリ内の前記データを検出し、前記第1の処理ユニットによる介入なしに前記IDを用いて、前記データを前記宛先に転送する第2の処理ユニットと、
を備えていることを特徴とするチップ。

9. 請求項8記載のチップにおいて、前記第2の処理ユニットは、前記データを前記宛先によって用いられるフォーマットに配置することを特徴とするチップ。

10. 請求項9記載のチップにおいて、前記ホストから受信されたデータは第1のフォーマットを有し、前記データは前記宛先によって用いられる第2のフォーマットにフォーマットされることを特徴とするチップ。

11. 請求項10記載のチップにおいて、前記第1のフォーマットはSCSIプロトコルであることを特徴とするチップ。

12. 請求項10記載のチップにおいて、前記第2のフォーマットはファイバ

・チャンネル・プロトコルであることを特徴とするチップ。

13. 請求項8記載のチップにおいて、前記第2の処理ユニットは、前記宛先へのデータ転送を制御するコンテキスト・マネージャを含み、このチップ

は、更に、前記コンテキスト・マネージャによって制御されデータを前記宛先に転送する送信機を備えていることを特徴とするチップ。

14. 請求項13記載のチップにおいて、
データを宛先から受信する受信機を更に備えており、

前記第2の処理ユニットは、前記データの受信を制御して、前記ホストへのデータの存在を前記第1の処理ユニットに指示することを特徴とするチップ。

15. 請求項8記載のチップにおいて、前記第2の処理ユニットは埋め込み型プロセッサであることを特徴とするチップ。

16. 請求項8記載のチップにおいて、前記第2の処理ユニットはステート・マシンであることを特徴とするチップ。

17. 請求項8記載のチップにおいて、前記第1の処理ユニットはプロトコル・エンジンであることを特徴とするチップ。

18. 請求項8記載のチップにおいて、前記第1の処理ユニットは、送信されるデータのリストを作成し、前記リストは、前記第2の処理ユニットによってデータを前記宛先に転送するのに用いられることを特徴とするチップ。

19. 請求項8記載のチップにおいて、

複数のフレームを受信する受信機を更に備えており、

前記第2の処理ユニットは、前記複数のフレームのそれぞれを、それらが前記受信機によって受信される際に処理し、前記複数のフレームのすべてがいつ受信されたかを判断して、前記複数のフレームのすべてが受信されたという指示を提供することを特徴とするチップ。

20. チップであって、

データを宛先に転送するというホストからのリクエストを受信するプロトコル・エンジンと、

前記プロトコル・エンジンに接続されており、前記ホストからの前記データを

受信するデータ・ムーバと、

前記プロトコル・エンジンと前記データ・ムーバとに接続されており、前記データ・ムーバによって受信されたデータを前記プロトコル・エンジンに

よる介入なしに前記宛先に転送するトランスポート制御ユニットと、
を備えていることを特徴とするチップ。

21. 請求項20記載のチップにおいて、前記データ・ムーバはバスによって前記ホストに接続されていることを特徴とするチップ。

22. 請求項20記載のチップにおいて、前記トランスポート制御ユニットは、

前記データ・ムーバに接続されており、前記ホストから受信されたデータを前記宛先に送信する送信機と、

前記データ・ムーバに接続されており、ソースから前記ホストへのデータを受信する受信機と、

前記受信機と前記送信機とに接続されており、前記送信機によるデータの送信と前記受信機によるデータの受信とを制御する処理ユニットと、
を含むことを特徴とするチップ。

23. 請求項22記載のチップにおいて、前記処理ユニットは埋め込み型プロセッサであることを特徴とするチップ。

24. ホストから宛先にデータを転送する装置であって、

第1のプロセッサであって、

前記ホストから前記宛先にデータを転送するリクエストを受信する第1の受信手段であって、前記リクエストは、それぞれが、前記宛先のIDを含む、第1の受信手段と、

前記宛先に転送するデータを受信する第2の受信手段と、

前記宛先と第1のプロトコルを有する前記データとをメモリに記憶する記憶手段と、

を含む第1のプロセッサと、

第2のプロセッサであって、

第1のフォーマットの前記データを第2のフォーマットにフォーマットしフォーマットされたデータを形成するフォーマット手段と、

前記フォーマットされたデータを前記記憶手段に記憶された前記宛先を用いて転送する転送手段と、

を含む第2のプロセッサと、

を備えていることを特徴とする装置。

25. 請求項24記載の装置において、前記第1のプロトコルはSCSIプロトコルであることを特徴とする装置。

26. 請求項24記載の装置において、前記第2のプロトコルはファイバ・チャネル・プロトコルであることを特徴とする装置。

27. 請求項24記載の装置において、前記リクエストは、ホスト・プロセッサがデータ転送のためのリクエストを配置するリクエスト・キューを用いてなされることを特徴とする装置。

28. 請求項24記載の装置において、前記第1のプロセッサは、前記宛先に送られるデータのリストを作成するリスト化手段を更に含み、前記第2のプロセッサにおける前記転送手段は、前記リストを用いてデータを前記宛先に転送することを特徴とする装置。

29. 請求項24記載の装置において、遠隔ソースからデータを受信するポートを更に含んでおり、前記第2のプロセッサは、

前記枝なくソースから受信された前記第2のプロトコルを有するデータを記憶する記憶手段と、

前記第2のプロトコルを有する前記データを前記第1のプロトコルにフォーマットする第2のフォーマット手段と、

前記第1のプロセッサにデータが前記遠隔ソースから受信されたことを指示する指示手段と、

を更に含むことを特徴とする装置。

30. チップであって、

ホスト上のバスからのデータを送信及び受信するバス・インターフェース・ユ

ニットと、

前記バス・インターフェース・ユニットに接続されており、前記バスに結合されたホスト・メモリからこのチップに結合されたローカル・メモリへの情報の転送を管理し、データをデバイスに送信するのに用いられた送信アクティビティのリストを発生するプロトコル・エンジンと、

前記デバイスへの通信リンクのためのインターフェースを提供するリンク・コントローラと、

前記リンク・コントローラに接続されており、前記リンク・コントローラによる前記デバイスへの転送のためのフォーマットへのデータのフォーマットを管理する送信機と、

前記リンク・コントローラに接続されており、前記リンク・コントローラによって受信されるデータを管理する受信機と、

前記プロトコル・エンジンに結合されており、送信アクティビティの前記リストを検討し、前記送信機を用いて前記リストに基づきデータの転送を実行して、前記受信機によって受信されるデータを処理するコンテキスト・マネージャと、
を備えていることを特徴とするチップ。

31. 請求項30記載のチップにおいて、前記プロトコル・エンジンは埋め込み型プロセッサであることを特徴とするチップ。

32. 請求項31記載のチップにおいて、前記コンテキスト・マネージャは埋め込み型プロセッサであることを特徴とするチップ。

33. 請求項31記載のチップにおいて、前記コンテキスト・マネージャはステート・マシンであることを特徴とするチップ。

34. 請求項30記載のチップにおいて、前記データはフレームの形式であり、前記コンテキスト・マネージャは異なるソースから順不同で受信されるフレームを処理することを特徴とするチップ。

35. 請求項30記載のチップにおいて、前記コンテキスト・マネージャは、送信アクティビティの前記リストを検討し、送信アクティビティの前記リストから第1の項目を読み出すことによってデータ転送を実行し、前記項目に対するデー

タがローカル・データであるかどうかを判断し、前記データがローカルであるという判断に回答して、前記データを前記ローカル・メモリから前記送信機に書き込むことを特徴とするチップ。

36. 請求項35記載のチップにおいて、前記コンテキスト・マネージャは、前記項目に対するデータがシステム・データであるという判断に回答して、

バスを介して、前記ホスト・メモリからのデータを送信機にロードすることを特徴とするチップ。

37. 請求項36記載のチップにおいて、前記送信機は、データとコンテキスト情報とを記憶する記憶装置と、前記記憶装置に接続されたフレームタを含むことを特徴とするチップ。

38. 請求項35記載のチップにおいて、送信アクティビティの前記リストは、複数の送信制御ブロックであることを特徴とするチップ。

39. チップであって、

第1のバスからホストへデータを送信及び受信するインターフェースを有するバス・インターフェース・ユニットと、

前記バス・インターフェース・ユニットに接続されており、ホスト・メモリからこのチップに結合されたメモリへの情報の転送を管理し、データ転送のリクエストに回答し、複数の動作モードを有するプロトコル・エンジンであって、

前記プロトコル・エンジンがホスト・メモリの中にあるデータ・ブロックを転送するリクエストを検出する第1の動作モードと、

前記リクエストの検出に回答する第2の動作モードであって、前記プロトコル・エンジンは、前記データ・ブロックを前記ホスト・メモリからローカル・メモリの中に移動させ、前記送信ブロックは、前記データ・ブロックと前記データ・ブロックを送信するのに用いられる情報とを含む、第2の動作モードと、
を備えたプロトコル・エンジンと

前記送信ブロックを検出し、前記送信ブロックを送信機に送信する転送エンジンであって、前記送信機は、前記送信ブロックを受信してデータ・ブロックを送信のためのフォーマットに配置し、前記フォーマットは前記情報から識別される

、転送エンジンと、

前記送信機に接続されており、前記デバイスへの通信リンクのためのインターフェースを提供するリンク・コントローラと、
を備えていることを特徴とするチップ。

40. 請求項39記載のチップにおいて、前記コンテキスト情報は、宛先への経路のIDを含むことを特徴とするチップ。

41. 請求項39記載のチップにおいて、前記リンク・コントローラは、ファイバ・チャンネル仲裁ループによる通信のために構成されていることを特徴とするチップ。』

【国際調査報告】

INTERNATIONAL SEARCH REPORT

International Application No.

PCT/US 99/06772

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 H04L29/06 G06F13/12

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED
Minimum documentation searched (classification system followed by classification symbols)
IPC 6 H04L G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the International search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

| Category * | Classification of document, with indication, where appropriate, of the relevant passages | Relevant to claim no. |
|------------|---|-----------------------|
| X | OLOGIC CORPORATION: "ISP2100 Intelligent Fibre Channel Processor" DATA SHEET, 'Online! 29 July 1997 (1997-07-29), XP002111444 Retrieved from the Internet: <URL:http://qlogic.qlc.com/products/pdf/data_sheets/83210-580-00c.pdf> 'retrieved on 1999-08-04! the whole document | 1-5 |
| X | SMITH J A ET AL: "TACHYON: A GIGABIT FIBRE CHANNEL PROTOCOL CHIP" HENLETT-PACKARD JOURNAL, vol. 47, no. 5, 1 October 1996 (1996-10-01), pages 99-112, XP000631672 page 101, left-hand column, line 6 - page 111, left-hand column, line 24 | 1-3, 5 |

-/-

☒ Further documents are listed in the continuation of page C☐ Patent family members are listed in annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not

considered to be of particular relevance

"E" earlier document but published on or after the international

filing date

"L" document which may throw doubts on priority claim(s) or

which is cited to establish the publication date of another

claim or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or

other means

"P" document published prior to the international filing date but

later than the priority date claimed

"T" later document published after the international filing date

or priority date and not in conflict with the application but

cited to understand the principle or theory underlying the

invention

"X" document of particular relevance, the claimed invention

cannot be considered novel or cannot be considered to

involve an inventive step when the document is taken alone

"Y" document of particular relevance, the claimed invention

cannot be considered to involve an inventive step when the

document is combined with one or more other such docu-

ments, such combination being obvious to a person skilled

in the art

"Z" document member of the same patent family

Date of the actual completion of the international search

Date of mailing of the international search report

5 August 1999

17/08/1999

Name and mailing address of the ISA
European Patent Office P B 5518 Patentstein 2
NL - 2000 HV - Rijswijk
Tel. (+31-70) 340-2000, Telex 31 05 1 wpa of
Fax (+31-70) 340-3018

Authorized officer

Eraso Helguera, J

INTERNATIONAL SEARCH REPORT

Inventor:
 Application No.
 PCT/US 99/06772

| C (Continuations) DOCUMENTS CONSIDERED TO BE RELEVANT | | |
|---|---|-----------------------|
| Category * | Citation of document, with indication, where appropriate, of the relevant passages | Relevant to claim No. |
| A | <p>CHILD J: "120 GEARS UP FOR EMBEDDED USE" COMPUTER DESIGN, vol. 36, no. 8, 1 August 1997 (1997-08-01), page 15/16, 18 XP000735779 ISSN: 0010-4566 page 18, right-hand column, paragraph 2</p> | 4 |

